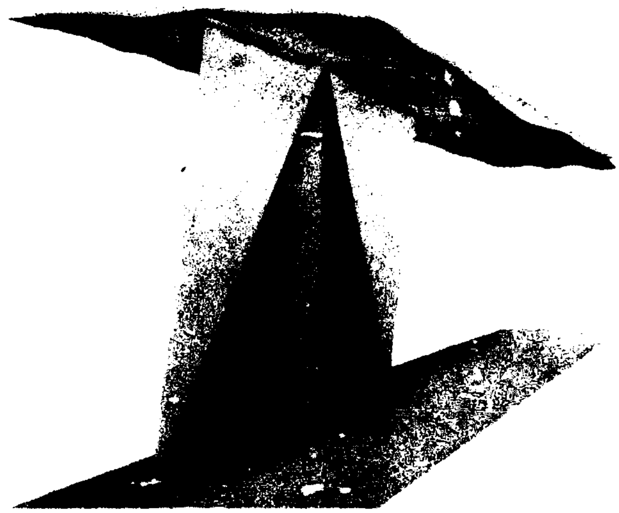
(4)

Technical Document 1275
April 1988

ELF Scattering in the Earth-Ionosphere Waveguide

R. A. Pappert
L. R. Hitney



Naval Ocean Systems Center
San Diego, California 92152-5000

NAVAL OCEAN SYSTEMS CENTER
San Diego, California 92152-5000

E. G. SCHWEIZER, CAPT, USN
Commander

R. M. HILLYER
Technical Director

ADMINISTRATIVE INFORMATION

The work reported herein was conducted for the Defense Nuclear Agency over the period January to August 1988.

Released by
J.A. Ferguson, Head
Modeling Branch

Under authority of
J.H. Richter, Head
Ocean and Atmospheric
Sciences Division

AS

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

ADA198948

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED

1b. RESTRICTIVE MARKINGS:

2a. SECURITY CLASSIFICATION AUTHORITY:

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE:

3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S): NOSC Technical Document 1275

5. MONITORING ORGANIZATION REPORT NUMBER(S):

6a. NAME OF PERFORMING ORGANIZATION: Naval Ocean Systems Center

6b. OFFICE SYMBOL (if applicable): NOSC

7a. NAME OF MONITORING ORGANIZATION: Naval Ocean Systems Center

6c. ADDRESS (City, State and ZIP Code): San Diego, California 92152-5000

7b. ADDRESS (City, State and ZIP Code): San Diego, California 92152-5000

8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Defense Nuclear Agency

8b. OFFICE SYMBOL (if applicable): DNA-RAAE

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER:

8c. ADDRESS (City, State and ZIP Code):
Radiation Directorate
Hybea Valley Federal Building
Washington, DC 20305

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	AGENCY ACCESSION NO.
62715H	599QMXBB	544-MP20	DN651 524

11. TITLE (include Security Classification): ELF SCATTERING IN THE EARTH-IONOSPHERE WAVEGUIDE

12. PERSONAL AUTHOR(S): R.A. Pappert and L.R. Hitney

13a. TYPE OF REPORT: Interim

13b. TIME COVERED: FROM Jan 1988 TO Aug 1988

14. DATE OF REPORT (Year, Month, Day): April 1988

15. PAGE COUNT: 132

16. SUPPLEMENTARY NOTATION:

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number):
Radio propagation
atmospheric effects
weapons effects

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Slab-model implementation of ELF scattering by a cylindrically symmetric but radially varying inospheric disturbance is explored. The ground-based transmitter and receiver may be located beneath the disturbance or outside its projection on the ground. Amplitudes and phases of the laterally dependent part of all transverse magnetic (TM) field components are calculated as a function of range for a given location of the disturbance or as a function of disturbance location for a fixed range. Although partial sum convergence difficulties limit its general utility, the resulting program can be used for a variety of case studies. Examples given include drifting sporadic-E patches, a nuclear depression, and a weak solar proton event.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED

22a. NAME OF RESPONSIBLE PERSON: R.A. Pappert

22b. TELEPHONE (include Area Code): 619-553-3069

22c. OFFICE SYMBOL: Code 542

DD FORM 1473, 84 JAN

83 APR EDITION MAY BE USED UNTIL EXHAUSTED
ALL OTHER EDITIONS ARE OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

CONTENTS

TABLES

ILLUSTRATIONS

I. INTRODUCTION

Since an ELF signal from a remote transmitter is received over a wide range of azimuth angles, lateral ionospheric gradients produced by, for example, nuclear depressions can produce significant effects on propagation in the lower ELF band. This is because the Fresnel zone size can be comparable to the transverse dimensions over which the disturbed ionosphere changes significantly. It has been common practice to estimate off-path effects by using a simple surface-propagation model introduced by Wait [1964], which is predicated on the assumption that the field can be separated into lateral and height-dependent factors. A number of workers have extended and used this model to predict the vertical electric field disturbance produced by lateral gradients [Field and Joiner 1979, 1982; Pappert, 1980; Ferguson, Hitney and Pappert, 1982; Shellman, 1983]. These studies involve either the numerical solution of an integral equation or numerical solution of the two-dimensional wave equation for the vertical E field. The methods are quite general in the sense that rather arbitrarily shaped disturbances can be modeled. This study is restricted to cylindrically symmetric disturbances, and the formalism of Greifinger and Greifinger [1977] is explored for the purpose of calculating both the vertical electric and the horizontal magnetic field components of the RF wave. This is of significance in ELF studies since it is generally the RF magnetic field component normal to the great circle path joining transmitter and receiver which is measured. The Greifingers' formulation is in terms of a single-component vector potential from which all field components are obtained as derivatives. The vector potential is expressible in terms of an infinite sum of products of cylindrical and harmonic functions, and so derivatives are also expressible as infinite sums of well-known functions. A major purpose of this study is to examine the practicality of calculating those sums for a variety of conditions.

Common to all of the cited works, as well as to the present study, is the neglect of excitation factor and other height-gain effects so that disturbance effects reflect solely the laterally dependent part of the fields. Common also to all of the works is the assumption that the single nonevanescent ELF mode is pure TM. This is a reasonable approximation under ambient and depressed ionospheric conditions but certainly can be more serious for sporadic-E type environments [Pappert and Moler, 1978]. The current study is also based on a flat-earth formulation and therefore is restricted to ranges $\lesssim$ 4 Mm, where, at the outer limit, errors of the order of a few tenths of a dB would be anticipated. Shellman's [1983] work is unique in that it does make full allowance for spherical spreading.

Essential theoretical background is given in the following section. In particular, radial variations of the cylindrically symmetric disturbances are approximated by a slab model. In this respect the present program is a generalization of the program used to generate the results in Pappert [1985]. A shortcoming of the program is that Graf's addition theorem for Bessel functions [Watson, 1952], which forms the basis of the calculation, diverges when the distances from the center of the disturbance to transmitter and receiver are equal. For some configurations, series convergence for extended ranges about those points can be difficult or impossible to achieve. Nonetheless, the program can, with care, be used for a wide variety of conditions to gain insights into the effects of lateral gradients of ELF propagation.

Program input and output is described in sections III and IV, respectively. The program requires eigenangle inputs for both the ambient and disturbed regions of the guide. These must be supplied from a waveguide program such as that of Morfitt and Shellman [1976]. Provision is made for plotting

amplitudes and phases of the ratio of disturbed fields to undisturbed fields as a function of range for fixed location of the disturbance or for fixed ranges as a function of location of the center of the disturbance along a straight line. As illustrations of possible applications of the present program, results for a sporadic-E type environment, for a nuclear burst, and for a weak solar proton event are given in section V. A program listing is given in Appendix A and the plotting program, apart from the Hewlett Packard International Standard Plotting Package (HPISPP) components, is given in Appendix B.

II. SUMMARY OF EQUATIONS

Consistent with the simplified surface propagation model, as developed by the Greifingers, it is assumed that the single nonevanescent modal field components which propagate within the earth-ionosphere waveguide at ELF frequencies are derivable from a vector potential which has only a vertical component which can be factored into a lateral and a height-dependent component. In the following, only the behavior of the lateral component will be dealt with so that excitation factor and height-gain effects are ignored. In source-free regions, the lateral component, g, of the vector potential satisfies the reduced two-dimensional wave equation,

$$\nabla^2 g(\rho,\phi)+k^2 S^2(\rho,\phi)g(\rho,\phi) = 0, \tag{1}$$

where

$$\nabla^2 = \frac{1}{\rho}\frac{\partial}{\partial\rho}\left(\rho\frac{\partial}{\partial\rho}\right) + \frac{1}{\rho 2}\frac{\partial^2}{\partial\phi^2}. \tag{2}$$

The free-space wave number is denoted by k and S in the sine of the eigenangle characterizing the single nonevanescent ELF mode. The polar coordinates ρ and ϕ are shown in the scattering diagram of Figure 1. Earth-curvature effects are ignored, so that Figure 1 represents a plan view in a flat-earth geometry. In the present work, S is taken to be a function of ρ only so that the disturbance is cylindrically symmetric. A slab approximation will be used to model the radial, or ρ, variation. Thus, the disturbance will be modeled as:

$$
\left.\begin{array}{ll}
S^2 = S_1^2 & \rho \leq \rho_1 \\
S^2 = S_2^2 & \rho_1 < \rho \leq \rho_2 \\
\vdots & \\
S^2 = S_n^2 & \rho_{n-1} < \rho \leq \rho_n \\
\vdots & \\
S^2 = S_{M-1}^2 & \rho_{M-2} < \rho \leq \rho_{M-1} \\
S^2 = S_M^2 & \rho_{M-1} < \rho
\end{array}\right\} , \tag{3}
$$

where S_n is the eigenvalue for the n^{th} slab, M is the total number of slabs, and S_M is the ambient eigenvalue.

The transmitter is taken to be a horizontal electric dipole located at the origin of Figure 1 and oriented at an angle δ relative to the x-axis. The receiver is taken to be located along the x-axis at a distance r from the transmitter. The disturbance is centered a distance r_o from the transmitter. The incident wave associated with the dipole source is derivable from the lateral potential function (a time dependence $\exp(i\omega t)$ is assumed throughout this work).

$$
g_i = \cos(-\delta) H_1^{(2)}(kS_T r), \tag{4}
$$

where $H_1^{(2)}$ is the Hankel function of the second kind of order 1 and the subscript T on S denotes the slab in which the transmitter is located. In this

connection, the subsequent development allows for the transmitter to be beneath the disturbance or outside its projection on the ground.

Once the lateral potential function, g, is determined, the laterally dependent parts of the ground magnetic field components H_ρ and H_ϕ are given by

$$H_\rho = \frac{1}{\rho}\frac{\partial g}{\partial \phi}, \quad H_\phi = -\frac{\partial g}{\partial \rho}. \tag{5}$$

Also, to good approximation, the laterally dependent part of the ground vertical electric field is given by

$$E_z = -i\omega S_R^2 g, \tag{6}$$

where the subscript R on S signifies the slab in which the receiver is located. The subsequent development also allows for the receiver to be beneath the disturbance or outside its projection on the ground.

The total lateral potential is determined by adding to g_i the scattered component which satisfies Equation (1) and by requiring E_z and H_ϕ to be continuous at each slab boundary. It is remarked that the scattered component of g represents the total component in all slabs except the slab containing the transmitter.

The scattered component of g in slabs 1, $n \neq 1$, M, and M is written as (M is the total number of slabs):

i) $n = 1, \ \rho \leq \rho_1$

$$g_s = \sum_m \left\{c_{jm}^{(1)} \cos m\phi + s_{jm}^{(1)} \sin m\phi\right\} J_m(kS_1\rho) \tag{7}$$

ii) $n \neq 1, M,\ \rho_{n-1} < \rho \leq \rho_n$

$$g_s = \sum_m \left\{c_{jm}^{(n)} \cos m\phi + s_{jm}^{(n)} \sin m\phi\right\} J_m(kS_n\rho)$$

$$+ \sum_m \left\{c_{hm}^{(n)} \cos m\phi + s_{hm}^{(n)} \sin m\phi\right\} H_m^{(2)}(kS_n\rho) \tag{8}$$

iii) $n = M,\ \rho_m < \rho$

$$g_s = \sum_m \left\{c_{hm}^{(n)} \cos m\phi + s_{hn}^{(n)} \sin m\phi\right\} H_m^{(2)}(kS_n\rho). \tag{9}$$

The $c_{jm}^{(n)}$'s and $s_{jm}^{(n)}$'s are constants to be determined by the boundary conditions and the J_m's ($H_m^{(2)}$'s) are Bessel functions (Hankel functions of the second kind). The sum over m runs from zero to infinity. Continuity of the ratio E_z/H_ϕ at the slab boundaries yields (T is the slab containing the transmitter, and M is the outermost slab):

$$\gamma_m^{(2)} = \frac{c_{hm}^{(2)}}{c_{jm}^{(2)}} = \frac{s_{hm}^{(2)}}{s_{jm}^{(2)}}$$

$$= \frac{S_2 J_m'(kS_1\rho_1) J_m(kS_2\rho_1) - S_1 J_m(kS_1\rho_1) J_m'(kS_2\rho_1)}{S_1 J_m(kS_1\rho_1) H_m^{(2)\prime}(kS_2\rho_1) - S_2 J_m'(kS_1\rho_1) H_m^{(2)}(kS_2\rho_1)}, \quad 3 \le T \le M \qquad (10)$$

$$\gamma_m^{(n)} = \frac{c_{hm}^{(n)}}{c_{jm}^{(n)}} = \frac{s_{hm}^{(n)}}{s_{jm}^{(n)}} = \frac{p - q}{u - v}, \quad 4 \le T \le M, \ 2 < n < T, \qquad (11)$$

where

$$p = S_n J_m(kS_n\rho_{n-1}) \left[J_m'(kS_{n-1}\rho_{n-1}) + \gamma_m^{(n-1)} H_m^{(2)\prime}(kS_{n-1}\rho_{n-1}) \right] \qquad (12)$$

$$q = S_{n-1} J_m'(kS_n\rho_{n-1}) \left[J_m(kS_{n-1}\rho_{n-1}) + \gamma_m^{(n-1)} H_m^{(2)}(kS_{n-1}\rho_{n-1}) \right] \qquad (13)$$

$$u = S_{n-1} H_m^{(2)\prime}(kS_n\rho_{n-1}) \left[J_m(kS_{n-1}\rho_{n-1}) + \gamma_m^{(n-1)} H_m^{(2)}(kS_{n-1}\rho_{n-1}) \right] \qquad (14)$$

$$v = S_n H_m^{(2)}(kS_n\rho_{n-1}) \left[J_m'(kS_{n-1}\rho_{n-1}) + \gamma_m^{(n-1)} H_m^{(2)\prime}(kS_{n-1}\rho_{n-1}) \right]. \qquad (15)$$

The primes occurring in the preceding equations represent derivatives with respect to the arguments.

For $n > T$, continuity of the ratio of E_z/H_ϕ at the slab boundaries yields

$$\Gamma_m^{(M-1)} = \frac{c_{jm}^{(M-1)}}{c_{hm}^{(M-1)}} = \frac{s_{jm}^{(M-1)}}{s_{hm}^{(M-1)}}$$

$$= \frac{S_{M-1}H_m^{(2)\prime}(kS_M\rho_{M-1})H_m^{(2)}(kS_{M-1}\rho_{M-1}) - S_M H_m^{(2)}(kS_M\rho_{M-1})H_m^{(2)\prime}(kS_{M-1}\rho_{M-1})}{S_M H_m^{(2)}(kS_M\rho_{M-1})J_m'(kS_{M-1}\rho_{M-1}) - S_{M-1}H_m^{(2)\prime}(kS_M\rho_{M-1})J_M(kS_{M-1}\rho_{M-1})}, 1 \le T \le M-2 \quad (16)$$

$$\Gamma_m^{(n)} = \frac{c_{jm}^{(n)}}{c_{hm}^{(n)}} = \frac{s_{jm}^{(n)}}{s_{hm}^{(n)}} = \frac{p1 - q1}{u1 - v1},\ 1 \le T \le M - 3,\ T < n < M - 1, \quad (17)$$

where

$$p1 = S_n H_m^{(2)}(kS_n\rho_n)\left[\Gamma_m^{(n+1)}J_m'(kS_{n+1}\rho_n) + H_m^{(2)\prime}(kS_{n+1}\rho_n)\right] \quad (18)$$

$$q1 = S_{n+1}H_m^{(2)\prime}(kS_n\rho_n)\left[\Gamma_m^{(n+1)}J_m(kS_{n+1}\rho_n) + H_m^{(2)}(kS_{n+1}\rho_n)\right] \quad (19)$$

$$u1 = S_{n+1}J_m'(kS_n\rho_n)\left[\Gamma_m^{(n+1)}J_m(kS_{n+1}\rho_n) + H_m^{(2)}(kS_{n+1}\rho_n)\right] \quad (20)$$

$$v1 = S_n J_m(kS_n\rho_n)\left[\Gamma_m^{(n+1)}J_m'(kS_{n+1}\rho_n) + H_m^{(2)\prime}(kS_{n+1}\rho_n)\right]. \quad (21)$$

The choice of defining γ and Γ by their respective ratios has been found to yield the best numerical stability when using an L-U decomposition algorithm for solving the linear equations imposed by the continuity of E_z and H_ϕ at the slab boundaries ρ_{T-1} and ρ_T. The conditions imposed on Equations (10), (11), (16), and (17) show that for M = 2, neither γ's nor Γ's need be calculated. In that case, there are only two unknowns ($c_{jm}^{(1)}$, $c_{hm}^{(2)}$) associated with the cosine terms for each m and two unknowns ($s_{jm}^{(1)}$, $s_{hm}^{(2)}$) associated with the sine terms for each m. Each pair in this instance is calculated from the condition

of continuity of E_z and H_ϕ at ρ_1. The continuity equations at ρ_{T-1} and ρ_T are established by noting that the incident lateral potential functions for radiation at an angle α relative to the x-axis is:

$$g_i = \cos(\alpha-\delta)H_1^{(2)}(kS_T r). \tag{22}$$

By using Graf's theorem [Watson, 1952], g_i may be expanded as follows:

i) $\rho < r_o$

$$g_i = -2\cos(\theta-\delta)\sum_m \epsilon_m H_m^{(2)\prime}(kS_T r_o)J_m(kS_T\rho)\cos m\phi$$

$$+ 2\sin(\theta-\delta)\sum_m mG_m(kS_T r_o)J_m(kS_T\rho)\sin m\phi$$

$$= \sum_m (p_{mc}^{(T)}\cos m\phi + p_{ms}^{(T)}\sin m\phi) \tag{23}$$

ii) $\rho > r_o$

$$g_i = -2\cos(\theta-\delta)\sum_m \epsilon_m J_m'(kS_T r_o)H_m^{(2)}(kS_T\rho)\cos m\phi$$

$$+ 2\sin(\theta-\delta)\sum_m mL_m(kS_T r_o)H_m^{(2)}(kS_T\rho)\sin m\phi$$

$$= \sum_m (q_{mc}^{(T)} \cos m\phi + q_{ms}^{(T)} \sin m\phi), \tag{24}$$

where

$$\epsilon_m = \begin{matrix} 0.5 & m = o \\ 1 & m \neq o \end{matrix} \tag{25}$$

$$G_m(x) = H_m^{(2)}(x)/x \tag{26}$$

$$L_m(x) = J_m(x)/x. \tag{27}$$

It is remarked that Equations (23) and (24) diverge for $r_o = \rho$, thereby restricting the generality of the development as mentioned in the introduction. By using these decompositions along with the formula for H_ϕ given in Equation (5) and the formula for E_z given in Equation (6), the continuity conditions at ρ_{T-1} and ρ_T yield for the coefficients of the cosine terms:

$$-S_{T-1}^2 c_{jm}^{(T-1)} \left[J_m(kS_{T-1}\rho_{T-1}) + \gamma_m^{(T-1)} H_m^{(2)}(kS_{T-1}\rho_{T-1})\right] + S_T^2 \left[c_{jm}^{(T)} J_m(kS_T\rho_{T-1}) + c_{hm}^{(T)} H_m^{(2)}(kS_T\rho_{T-1})\right]$$

$$= 2S_T^2 \cos(\theta - \delta)\epsilon_m H_m^{(2)\prime}(kS_T r_o) J_m(kS_T\rho_{T-1}) \tag{28}$$

$$-S_{T-1} c_{jm}^{T-1} \left[J_m'(kS_{T-1}\rho_{T-1}) + \gamma_m^{(T-1)} H_m^{(2)\prime}(kS_{T-1}\rho_{T-1})\right] + S_T \left[c_{jm}^{(T)} J_m'(kS_T\rho_{T-1}) + c_{hm}^{(T)} H_m^{(2)\prime}(kS_T\rho_{T-1})\right]$$

$$= 2S_T \cos(\theta - \delta)\epsilon_m H_m^{(2)\prime}(kS_T r_o) J_m'(kS_T\rho_{T-1}) \tag{29}$$

$$S_T^2\left[c_{jm}^{(T)}J_m(kS_T\rho_T)+c_{hm}^{(T)}H_m^{(2)}(kS_T\rho_T)\right]-S_{T+1}^2c_{hm}^{(T+1)}\left[\Gamma_m^{(T+1)}J_m(kS_{T+1}\rho_T)+H_m^{(2)}(kS_{T+1}\rho_T)\right]$$

$$= 2S_T^2\cos(\theta-\delta)\epsilon_m J_m'(kS_Tr_o)H_m^{(2)}(kS_T\rho_T) \tag{30}$$

$$S_T\left[c_{jm}^{(T)}J_m'(kS_T\rho_T)+c_{hm}^{(T)}H_m^{(2)\prime}(kS_T\rho_T)\right]-S_{T+1}c_{hm}^{(T+1)}\left[\Gamma_m^{(T+1)}J_m'(kS_{T+1}\rho_T)+H_m^{(2)\prime}(kS_{T+1}\rho_T)\right]$$

$$= 2S_T\cos(\theta-\delta)\epsilon_m J_m'(kS_Tr_o)H_m^{(2)\prime}(kS_T\rho_T). \tag{31}$$

The corresponding equations for the coefficients of the sine terms are:

$$-S_{T-1}^2s_{jm}^{(T-1)}\left[J_m(kS_{T-1}\rho_{T-1})+\gamma_m^{(T-1)}H_m^{(2)}(kS_{T-1}\rho_{T-1})\right]+S_T^2\left[s_{jm}^{(T)}J_m(kS_T\rho_{T-1})+s_{hm}^{(T)}H_m^{(2)}(kS_T\rho_{T-1})\right]$$

$$= -2S_T^2\sin(\theta-\delta)mG_m(kS_Tr_o)J_m(kS_T\rho_{T-1}) \tag{32}$$

$$-S_{T-1}s_{jm}^{(T-1)}\left[J_m'(kS_{T-1}\rho_{T-1})+\gamma_m^{(T-1)}H_m^{(2)\prime}(kS_{T-1}\rho_{T-1})\right]+S_T\left[s_{jm}^{(T)}J_m'(kS_T\rho_{T-1})+s_{hm}^{(T)}H_m^{(2)\prime}(kS_T\rho_{T-1})\right]$$

$$= -2S_T\sin(\theta-\delta)mG_m(kS_Tr_o)J_m'(kS_T\rho_{T-1}) \tag{33}$$

$$S_T^2\left[s_{jm}^{(T)}J_m(kS_T\rho_T)+s_{hm}^{(T)}H_m^{(2)}(kS_T\rho_T)\right]-S_{T+1}^2s_{hm}^{(T+1)}\left[\Gamma_m^{(T+1)}J_m(kS_{T+1}\rho_T)+H_m^{(2)}(kS_{T+1}\rho_T)\right]$$

$$= -2S_T^2\sin(\theta-\delta)mL_m(kS_Tr_o)H_m^{(2)}(kS_T\rho_T) \tag{34}$$

$$S_T\left[s_{jm}^{(T)}J_m'(kS_T\rho_T)+s_{hm}^{(T)}H_m^{(2)\prime}(kS_T\rho_T)\right]-S_{T+1}s_{hm}^{(T+1)}\left[\Gamma_m^{(T+1)}J_m'(kS_{T+1}\rho_T)+H_m^{(2)\prime}(kS_{T+1}\rho_T)\right]$$

$$= -2S_T \sin(\theta-\delta) m L_m(kS_T r_o) H_m^{(2)\prime}(kS_T \rho_T). \quad (35)$$

In the most general case there are four equations and four unknowns. Special cases are:

$T = 1$: Use Equations (30) and (31) for cosine coefficients and set $C_{hm}^{1} = 0$. Use Equations (34) and (35) for sine coefficients and set $S_{hm}^{1} = 0$

$T = 2$: Set $\gamma_m^{(1)} = 0$.

$T = M-1$: Set $\Gamma_m^{(M)} = 0$.

$T = M$: Use Equations (28) and (29) for cosine coefficients and set $c_{jm}^{(M)} = 0$. Use Equations (32) and (33) for sine coefficients and set $s_{jm}^{(M)} = 0$.

It should be observed that if $M = 2$ or 3, two of the special cases can occur simultaneously. For example, if $M = 2$ and $T = 2$ then Equations (28) and (29) would be used for the cosine coefficients with $\gamma_m^{(1)}$ set to zero and Equations (32) and (33) would be used for the sine terms with $\gamma_m^{(1)}$ set to zero. After solving for the unknown coefficients in Equations (28) through (31) and (32) through (35), we obtain all remaining unknown coefficients from Equations (10), (11), (16), and (17). With the $c_{jm}^{(n)}$'s, $c_{hm}^{(n)}$'s, $s_{jm}^{(n)}$'s and $s_{hm}^{(n)}$'s known, the scattered part of the lateral potential function is given by Equations (7)

through (9). The latter represent the total potential in all slabs except that containing the transmitter. In that case, the total potential is given by the sum of the scattered potential and the primary lateral potential function given in Equation (4). To improve convergence properties in the neighborhood of $r_o \approx \rho$ for slabs $n \neq n_T$, it is preferable to write Equations (7) through (9) as:

i) $n = 1,\ \rho \leq \rho_1,\ n \neq n_T$

$$g_s = \sum_m \Bigg[\Big(c_{jm}^{(1)} J_m(kS_1\rho) - \begin{Bmatrix} p_{mc}^{(1)} \\ q_{mc}^{(1)} \end{Bmatrix}\Big) \cos m\phi$$

$$+ \Big(s_{jm}^{(1)} J_m(kS_1\rho) - \begin{Bmatrix} p_{ms}^{(1)} \\ q_{ms}^{(1)} \end{Bmatrix}\Big) \sin m\phi\Bigg] + \cos(-\delta) H_1^{(2)}(kS_1 r) \qquad (36)$$

ii) $n \neq 1, M,\ \rho_{n-1} < \rho \leq \rho_n,\ n \neq n_T$

$$g_s = \sum_m \Bigg[\Big(c_{jm}^{(n)} J_m(kS_n\rho) + c_{hm}^{(n)} H_m^{(2)}(kS_n\rho) - \begin{Bmatrix} p_{mc}^{(n)} \\ q_{mc}^{(n)} \end{Bmatrix}\Big) \cos m\phi$$

$$+ \Big(s_{jm}^{(n)} J_m(kS_n\rho) + s_{hm}^{(n)} H_m^{(2)}(kS_n\rho) - \begin{Bmatrix} p_{mc}^{(n)} \\ q_{ms}^{(n)} \end{Bmatrix}\Big) \sin m\phi\Bigg]$$

$$+ \cos(-\delta) H_1^{(2)}(kS_n r) \qquad (37)$$

iii) $n = M$, $\rho_M < \rho$, $n \neq n_T$

$$g_s = \sum_m \left[\left(c_{hm}^{(M)} H_m^{(2)}(kS_M\rho) - \begin{Bmatrix} p_{mc}^{(M)} \\ q_{mc}^{(M)} \end{Bmatrix} \right) \cos m\phi \right.$$

$$\left. + \left(s_{hm}^{(M)} H_m^{(2)}(kS_M\rho) - \begin{Bmatrix} p_{ms}^{(M)} \\ q_{ms}^{(M)} \end{Bmatrix} \right) \sin m\phi \right] + \cos(-\gamma) H_1^{(2)}(kS_M r), \qquad (38)$$

where the p's and q's are defined in Equations (23) and (24). In particular, the p's are used if $\rho < r_o$ and the q's if $\rho > r_o$.

The RF field components are given by Equations (5) and (6). From Figure 1 it will be seen that the horizontal magnetic field components H_r and H_y are expressed as follows in terms of the components H_ρ and H_ϕ:

$$H_r = H_\rho \cos\chi + H_\phi \sin\chi \qquad (39)$$

$$H_y = -H_\phi \sin\chi + H_\rho \cos\chi . \qquad (40)$$

Amplitude behavior of $E_z/E_z{}^u$, $H_y/H_y{}^u$ and $H_r/H_r{}^u$ and phase behavior of their reciprocals are the principal program outputs, where the superscript u signifies the unperturbed value.

III. DESCRIPTION OF INPUT

Most input to the program is supplied in a namelist file of 16 or fewer characters. Examples of two such files are shown in Tables 1 and 2. Table 1 gives input for a range variation, and Table 2 provides an example for a disturbance variation. The namelist inputs are:

rho(mxslab)........................Cylindrical slab boundaries in km. Also, mxslab is a parameter setting the dimensioning for the number of slabs. The number of rho's required is one less than the number of slabs.

iflag.............................Integer flag. Range variation is given by iflag = 0, disturbance variation by iflag $\neq$ 0.

nrslab............................Integer value of number of cylindrical slabs.

rmin..............................Starting value in km for range variation when iflag = 0. When iflag $\neq$ 0 it is the range value in km for which calculations are performed as the disturbance moves.

rmax..............................Last range value calculated (in km) when iflag = 0. Not used otherwise.

Table 1. Sample input (range variation)

```
&datum
 rho=2750.,2800.,2850.,2900.,2950.,3000.,3050.,3100.,
3150.,3200.,
3250.,3300.,3350.,3400.,3450.,3500.,3550.,
3600.,3650.,3700.,
3750.,3800.,
 iflag=0,
 nrslab=23,
 rmin=100.,
 rmax=5000.,
 rinc=50.,
 smin=1802.78.,
 smax=40.,
 sinc=50.,
 alpha=0.,
 delta1=0.,
 delta2=90.,
 c1=(1.,0.),
 c2=(0.,1.),
 xyint=3000.,
 thta=(81.305,-40.807),(81.307,-40.801),(81.309,-40.796),(81.311,-40.784),
(81.317,-40.759),(81.331,-40.708),(81.358,-40.604),
(81.411,-40.395),
(81.515,-39.992),(81.703,-39.266),
(82.007,-38.105),(82.420,-36.560),
(82.861,-34.938),(83.231,-33.611),
(83.481,-32.727),(83.627,-32.218),
(83.704,-31.948),(83.743,-31.812),(83.763,-31.744),(83.772,-31.712),
(83.776,-31.696),(83.778,-31.688),(83.781,-31.681),
 frqkhz=.075,
 skip=220.,
&end
```

Table 2. Sample input (disturbance variation)

```
&datum
 rho=31.25,62.5,93.75,125.,156.25,187.5,218.75,250.,281.25,312.5,343.75,
375.,406.25,437.5,468.75,500.,
 iflag=1,
 nrslab=17,
 rmin=1600.,
 rmax=0.,
 rinc=50.,
 smin=-1200.,
 smax=-600.,
 sinc=100.,
 alpha=-57.44,
 delta1=0.,
 delta2=90.,
 c1=(1.,0.),
 c2=(0.,1.),
 xyint=1250.,
 thta=(59.392805,-65.552101),(60.929764,-63.640642),(62.466723,-61.729183),
(64.003682,-59.817724),(65.540642,-57.906265),(67.077601,-55.994806),
(68.61456,-54.083348),(70.151519,-52.171889),(71.688479,-50.26043),
(73.225438,-48.348971),(74.762397,-46.437512),(76.299356,-44.526054),
(77.836361,-42.614595),(79.373275,-40.703136),(80.910234,-38.791677),
(82.447193,-36.880218),(83.984153,-34.96876),
 frqkhz=.075,
 skip=50.,
&end
```

rinc.............................Range increment in km when iflag = 0. Not used otherwise.

smin.............................When iflag $\neq$ 0, smin is the starting value in km of disturbance center along s (see Figure 2). When iflag = 0 it is the value of the disturbance center along s for which range variations are performed.

smax.............................When iflag = 0 smax is the final value in km of disturbance center along s (see Figure 2). Not used otherwise.

sinc.............................Increment in km by which disturbance center is moved along s (see Figure 2). Not used otherwise.

alpha............................Angle in degrees denoting slope of straight line upon which the disturbance center lies (see Figure 2). If iflag = 0, the center is fixed. If iflag $\neq$ 0, the center moves along s. In the latter case the direction of motion is along positive s if alpha > 0 and along negative s if alpha $\leq$ 0.

delta1, delta2...................The program allows for two horizontal electric dipole sources at the origin. Delta1 and delta2 are the angles in degrees made by the dipoles with the x-axis (see

Figure 1). Crossed dipoles could have, for example, delta1 = 0°, delta2 = 90°.

c1, c2............................Complex amplitude factors of the two electric dipole sources. Dipoles 90° out of phase could have, for example, c1 = (1., 0.), c2 = (0.,1.).

xyint.............................If alpha $\neq$ 0., xyint is the intercept of the straight line s with the x-axis. In Figure 2 that intercept is denoted by D. If alpha = 0°, s is parallel to x, and xyint is the y-intercept of the path s. A given value of x0 and y0 can be set for range variation calculations by setting alpha = 90°. Then, x0 = xyint and y0 = smin.

thra(mxslab)......................Input eigenangles in degrees from, for example, a waveguide program such as that of Morfitt and Shellman [1976]. An eigenangle for each slab is required

frqkhz............................Radio frequency in kHz.

skip..............................Distance in km about $r_o = \rho$ (see Figure 1), for which calculations are skipped because of poor convergence of the Bessel and Hankel function expansions. The actual distance

skipped along s if iflag $\neq$ 0 or along r if iflag = 0 will depend upon the actual geometry.

A crucial value is mmax, which is set in a parameter statement in the subroutines mlinit, mlcoef, mlflds, and hjfunc. It controls the number of terms used in the Bessel and Hankel function expansions. In examples given in a later section of this report, mmax values of 50 and 60 have been used. Generally, increasing mmax allows the namelist input skip to be reduced only slightly, because convergence of the Bessel and Hankel function expansion is very slow in the neighborhood of $\rho \approx r_o$. For some variations overflow problems have occurred with mmax settings $\approx$ 80.

IV. DESCRIPTION OF OUTPUT

A hard copy of output is available in the file 'mlout.' It is particularly useful when concerned about sensitive areas of calculation which may critically depend upon the input skip and upon the parameter setting mmax. Table 3 shows output corresponding to the input of Table 2. The first section of output echoes the namelist input values. Next is a printout of mmax. Each section following that begins with the slab number containing the transmitter (nt) and the slab number containing the receiver (nr). Following that are the angles χ, Θ and ϕ in radians (see Figure 1) for the current geometry. Next comes the range (r) in km, the distance from transmitter to center of disturbance (r_o) in km and the distance from the center of the disturbance to the receiver (rhorcv - this is denoted by ρ in Figure 1) in km. Next come values for x0 and y0 in km, which are the x and y coordinates of the center of the disturbance (see Figure 1). Next the location of the center of the disturbance along the s axis (scoord) is given in km. Then output for the disturbed-to-ambient amplitude ratios in dB and their phase differences in degrees are given. In sensitive regions a convergence warning is given and a comparison is made between the partial sum decomposition of the fields (ezps, hyps, hrps) and their exact values (eztst, hytst, hrtst). These comparisons are made only as an indicator of possible problems. They compare the partial sum calculation with the exact value for the range r using the transmitter eigenangle. Generally, inadequacies of the calculations for particular ranges, etc., are indicated by erratic behavior of the plots generated by the plotting routine discussed subsequently. Observe that although sinc =100., scoord jumps from -1000. to -700. This is because skip = 50. and because $|r_o$-rhorcv$|<50.$ for scoord = -900. and -800. The curves generated by the plotting

```
$DATUM
RHO = 31.250000000000000, 62.500000000000000, 93.750000000000000,
125.0, 156.250000000000000, 187.500000000000000, 218.750000000000000,
250.0, 281.250000000000000, 312.500000000000000, 343.750000000000000,
375.0, 406.250000000000000, 437.500000000000000, 468.750000000000000,
500.0, 0.00000000000000E+000, 0.00000000000000E+000, 0.00000000000000E+000,
0.00000000000000E+000,
IFLAG = 1,
NRSLAB = 17,
RMIN = 1600.0,
RMAX = 0.00000000000000E+000,
RINC = 50.0,
SMIN = -1200.0,
SMAX = -600.0,
SINC = 100.0,
ALPHA = -57.439999999999998,
DELTA1 = 0.00000000000000E+000,
DELTA2 = 90.0,
C1 = (1.0,0.00000000000000E+000),
C2 = (0.00000000000000E+000,1.0),
XYINT = 1250.0,
THTA = (59.392805000000003,-65.552100999999993),
 (60.929763999999999,-63.640642000000000), (62.466723000000002,
-61.729182999999999), (64.003681999999998,-59.817723999999998),
 (65.540642000000005,-57.906264999999998), (67.077601000000001,
-55.994805999999997), (68.614559999999997,-54.083348000000001),
 (70.151518999999993,-52.171889000000000), (71.688479000000001,
-50.260429999999999), (73.225437999999997,-48.348970999999999),
 (74.762396999999993,-46.437511999999998), (76.299356000000003,
-44.526054000000002), (77.836360999999997,-42.614595000000001),
 (79.373275000000007,-40.703136000000001), (80.910234000000003,
-38.791677000000000), (82.447192999999999,-36.880217999999999),
 (83.984153000000006,-34.968760000000003), (0.00000000000000E+000,
0.00000000000000E+000), (0.00000000000000E+000,0.00000000000000E+000),
 (0.00000000000000E+000,0.00000000000000E+000),
FRQKHZ = 7.50000000000000E-002,
SKIP = 50.0
$end
mmax= 51
nt= 17nr= 17
chi= .793157573897528theta= 1.032309585380254phi= 1.316125581734791
r= 1600.0r0= 1178.113912735650000rhorcv= 1419.356660749240700
x0= 604.180956552521020y0= 1011.393970280910480
scoord= -1200.0
ez/ezu(db)= -.165506486442835
ez/ezu(deg)= 1.681470061692652
hy/hyu(db)= -3.94426540952422E-002
hy/hyu(deg)= 2.882621844622661
hr/hru(db)= -.583301884765237
hr/hru(deg)= -3.085520568375234
nt= 17nr= 17
chi= .777432256265729theta= .953569521683634phi= 1.410590963063211
r= 1600.0r0= 1136.880831676240600rhorcv= 1321.703655468501400
x0= 657.999210173144320y0= 927.111139424167960
scoord= -1100.0
ez/ezu(db)= -.252368993800051
ez/ezu(deg)= 2.274333107647623
hy/hyu(db)= -6.70531978625971E-002
```

```
hy/hyu(deg)= 3.439969724323339
hr/hru(db)= -.939223897961050
hr/hru(deg)= -3.454996754144538
nt= 17nr= 17
chi= .759203236734368theta= .869470069855074phi= 1.512919434423131
r= 1600.0r0= 1103.197017528790900rhorcv= 1224.429571410443700
x0= 711.817463793767620y0= 842.828308567425440
convergence warning
eztst= (.379935937456921,-.415160527702075)ezps=
(.380245644175559,-.416776157476905)
hytst= (-.270317510748229,.384160851519301)hyps=
(-.242660137752635,.370969437093575)
hrtst= (-.119815092087581,.104044376392299)hrps=
(-.106669605168189,.131705939060073)
scoord= -1000.0
ez/ezu(db)= -.349150339481386
ez/ezu(deg)= 3.072486995179625
hy/hyu(db)= -.110864889350277
hy/hyu(deg)= 4.108500334753247
hr/hru(db)= -1.342326254355762
hr/hru(deg)= -3.661623642820115
nt= 17nr= 17
chi= .681913270322307theta= .594159466790211phi= 1.865520003900055
r= 1600.0r0= 1053.888306054818000rhorcv= 936.060597793248460
x0= 873.272224655637500y0= 589.979815997197870
convergence warning
eztst= (.379935937456921,-.415160527702075)ezps=
(.379354113228671,-.415575175498021)
hytst= (-.270317510748229,.384160851519301)hyps=
(-.271096227011937,.401539362971401)
hrtst= (-.119815092087581,.104044376392299)hrps=
(-.137190518301539,.103267986744807)
scoord= -700.0
ez/ezu(db)= -.688823339745501
ez/ezu(deg)= 6.319848973352530
hy/hyu(db)= -.405508957288416
hy/hyu(deg)= 6.410720265774057
hr/hru(db)= -2.577415646978328
hr/hru(deg)= -4.538105647620277
nt= 17nr= 17
chi= .644465709277133theta= .499356090273509phi= 1.997770941461931
r= 1600.0r0= 1056.042705429402500rhorcv= 841.746199995353440
x0= 927.090478276260800y0= 505.696985140455350
scoord= -600.0
ez/ezu(db)= -.821916249843746
ez/ezu(deg)= 7.486713866467464
hy/hyu(db)= -.588974755477543
hy/hyu(deg)= 7.033520953575946
hr/hru(db)= -2.858686768921191
hr/hru(deg)= -5.432117372794706
```

routine will be plotted by using simple linear interpolation between calculated points. Thus the values corresponding to scoord = -1000 and scoord = -700 will be joined by a straight line.

Another output of the program is an unformatted file termed "mlplot." This is used in conjunction with a plotting routine, which apart from components of HPISPP (Hewlett Packard International Standard Plotting Package) is given in Appendix B to plot any combination of disturbed-over-ambient amplitude ratio or phase differences for either range or source variation. The lateral field amplitude ratios are in dB, the phase differences are in degrees, and the range or source variation is in Mm. When executing the plot routine, it will first request:

Enter name of file containing data to be plotted.

After mlplot is entered, the screen will show:

Do you want amplitude plots?

Response: y or n

If the response is n, the questions will turn to phase plots. Otherwise questioning for amplitude plots will continue as follows:

How many amplitude plots do you want?

Response: 1, 2 or 3

If response is 1 or 2, the user will be instructed to:

Enter appropriate code for desired plot

if previous response was 1	}	1 ez plot 2 hy plot 3 hr plot
if previous response was 2	}	1 ez and hy plots 2 ez and hr plots 3 hy and hr plots

After the appropriate code (i.e., 1, 2 or 3) is entered, the screen will show (size x and size y are dimensions of x- and y-axis in inches):

```
Current values for size x and size y are 6.0 6.0
Do you want to change them?
Response:  y or n
```

If the response is y the screen will read:

```
Enter values for size x, size y
```

After size x and size y are entered, or if response to previous question was n, the screen will then read (xmin and xmax are the extreme left and right x coordinate values, respectively):

```
Current values for xmin and xmax are -5.0 7.0
Do you want to change them?
Response:  y or n
```

If the response is y the screen will read:

```
Enter values for xmin, xmax
```

After xmin and xmax are entered, or if response to previous question was n, the screen will then read (ymin and ymax are the extreme bottom and top y coordinate values, respectively):

Current values for ymin and ymax are -10.0 4.0

Do you want to change them?

Response y or n

If the response is y, the screen will read:

Enter value for ymin, ymax

After ymin and ymax are entered, or if response to previous question was n, the screen will then read:

Tic marks will be every 2.0 units on the x axis and every 2.0 units on the y-axis

Do you want to change them?

Response y or n

If the response is y, the screen will show:

Enter values for xtic and ytic

After xtic and ytic are entered, or if response to previous question was n, amplitude plots will then be started. Next inquiries about phase plots will appear beginning with

Do you want phase plots?

Response y or n

From this point on, the interrogation follows the same sequence described above for the amplitude plots. It asks how many phase plots are wanted, which ones, size of the x- and y-axes, coordinate bounds, and tic mark locations. With the exception of the slab variation sporadic-E results, the plots shown in the sample results section of this report were produced with the plotting package described above.

V. PROGRAM LAYOUT

This section describes only the basic features of the ELF propagation program. In particular, the plotting routine is not described beyond the discussion in the previous section.

The small driver program main opens the files mlout and mplot and controls the program flow. Subroutines in the order of their call are described below:

<u>subroutine mlinit</u>

Called from main, mlinit reads in namelist data. Quantities independent of transmitter and receiver locations are calculated in mlinit. These include the $\gamma_m^{(n)}$'s of Equations (10) and (11) and the $\Gamma_m^{(n)}$'s of Equations (16) and (17). Note that the $\Gamma_m^{(n)}$'s are called gammal(m,n) in the program.

<u>subroutine hjfunc</u> (arg, bj, bjp, h2, h2p)

Bessel functions, bj, and Hankel functions of the second kind, h2, and their derivatives bjp, h2p are generated in hjfunc for $m \leq$ mmax and complex argument, arg. If the percentage error in the Wronskian for m = 0 or m = mmax exceeds 0.01% a warning will be printed in mlout.

<u>subroutine cbesjy</u> (z, k, bj, by, kind, nprint)

Bessel functions of the first, bj, and second kind, by, of order k are calculated by standard power and asymptotic series for complex argument, z. The routine is used in this program only for $k \leq 2$ principally for the purpose of

obtaining starting values and as checks in the subroutine hjfunc. If the integer flag kind = 1, only Bessel functions of the first kind will be generated. Any other value of kind causes generation of Bessel functions of both kinds. The flag nprint = 0 causes no debug printout, whereas nprint = 1 will generate debug printout.

<u>subroutine mlcoef</u>

In this subroutine the unknown coefficients in Equations (28) through (31) and Equations (32) through (35) are determined by a Gaussian elimination process implemented in clineq. After the coefficients $c_{jm}^{(T)}$, $c_{jm}^{(T\pm1)}$, $c_{hm}^{(T)}$, $c_{hm}^{(T\pm1)}$, $s_{jm}^{(T)}$, $s_{jm}^{(T\pm1)}$, $s_{hm}^{(T)}$ and $s_{hm}^{(T\pm1)}$ have been determined for the transmitter slab T and its adjacent slabs, the coefficients for slabs $n \neq T$, $T \pm 1$ are determined by using Equations (10), (11), (16) and (17).

<u>subroutine clineq</u> (a, b, x, n, ndim, iflag, err)

The solutions of simultaneous linear equations with complex coefficients are determined by clineq. That is, it solves the matrix equation a*x=b for the vector x of length n, given the matrix a of size n x n and the vector b of length n by Crout's L-U decomposition. The matrix a is destroyed by clineq. The integer variable ndim must always be greater than or equal to n. The integer variable iflag has been set to zero in the present study. The real variable err is computed by clineq and indicates the relative error in the computed solution of vector x. Printout of err will occur only if it exceeds 10^{-5}.

subroutine mlflds

All quantities necessary for calculating the field components are available in this subroutine. For all slabs n for which $n \neq n_T$ the total lateral potential g equals the scattered component g_s. Thus when $n \neq n_T$ the field components are calculated from Equations (5) and (6) with $g = g_s$ and g_s given by Equations (36) through (38). When $n = n_T$, $g = g_s + g_i$, where g_i is the source field given by Equation (4). Magnetic field components H_y and H_r are calculated using the vector resolution given by Equations (39) and (40). Except for namelist and mmax, all printout onto the file 'mlout' occurs in the subroutine mlflds.

VI. ILLUSTRATIVE EXAMPLES

All results in this section are for crossed dipoles 90° out of phase. Figures 3 and 4 repeat results with the present program of results at 75 Hz obtained with an earlier program limited to two slabs [Pappert, 1985]. The radius of the disturbance is 500 km, and the disturbance models a resonant sporadic-E environment at about 120 km for which the attenuation rate is about 10 dB/Mm. The complex eigenangle for the disturbed region is (59.393°, -65.553°) and the complex eigenangle for the ambient region is (83.984°, -34.969°). Figure 3 gives amplitude behavior as a function of source location for the field components EZ, HY and HR at a range of 1.6 Mm as a function of source location along the path described by alpha = -57.44° and x intercept (D) = 600 km (see Figure 2). Figure 4 shows the phase behavior for the same disturbance and path of travel. The deep nulls in EZ and HY and their large phase excursions have been reconciled with Wisconsin test facility transmissions to Connecticut and the North Atlantic [Pappert, 1985]. The amplitude and phase behavior of HR does not, however, appear to be consistent with Bannister's [1986] radial magnetic field measurements. The reason for this is not clear, though it is suspected that it may be the result of TE model contamination which can occur under sporadic-E environments [Pappert and Moler, 1978].

To illustrate convergence with slab number it has been assumed that the sporadic-E environment of the previous paragraph is such that the eigenangles vary linearly with radius between the resonant disturbed value at the origin and the ambient value at the radius of 500 km. Figures 5 through 10 show amplitude and phase behaviors of all field components for slab numbers of 5, 9, and 17 as a function of source locations for the same path used in the previous paragraph. The curves indicate reasonable convergence for 17 slabs. By comparing with Figure 3, it will be seen that the amplitude fade for the

principal field components EZ and HY are reduced from about 8 dB to between 2 and 2.5 dB by virture of the linear variation of eigenangle. Similarly, comparison with Figure 4 shows that the phase advance for EZ and HY is reduced from more than 60° to between 15° and 20° because of the linear variation.

Further examples of the 17-slab model of the linearly varying sporadic-E environments of the previous paragraph are shown in Figures 11 through 18. Figures 11 and 12 give amplitude and phase results, respectively, for disturbance motion along a path which passes over the transmitter so that the slab index containing the transmitter changes. There is some evidence of this in the neighborhood of s ≃ -500 km and s ≃ 100 km, where some slight discontinuities are visible. Like the previous results the EZ and HY signatures are remarkably alike.

Figures 13 and 14 show amplitude and phase results, respectively, for disturbance motion which passes over the receiver. Unlike previous cases, EZ and HY amplitude and phase signatures are somewhat dissimilar. In particular, the HY amplitude fade is significantly deeper than the EZ fade, and the EZ fade advance is both larger and broader than that for HY.

Figures 15 and 16 give amplitude and phase results as a function of range variation. The disturbance center is located at x0 = 250 km and y0 = 0 km. It is believed that poor convergence of the partial sum expansions is responsible for the kinks which appear in the curves in the neighborhood of 700 km. In this connection it should be remarked that all of the results associated with the sporadic-E environments were obtained with skip = 50 km and mmax = 51.

Figures 17 and 18 also show amplitude and phase behavior as a function of range. The disturbance center in this instance is located at x0 = 3750 km and y0 = 0 km. The oscillatory behavior of the fades in the region between the transmitter and the disturbance is due to the standing wave pattern set up by the incident field and the field reflected by the disturbance. The ambient waveguide wavelength is 3374 km, and it will be seen that the periodicity of the standing wave pattern is, as expected, about one-half of that value. Also, the EZ and HY fades are dramatically different in this instance.

To illustrate another application of the program, Figures 19 through 26 show results for a nuclear environment analyzed by the Greifingers (1977). In particular they give approximate eigenvalues at 45 Hz for a bomb fission yield of 2 Mt as a function of ground truth, assuming that the fission products rose to 1000 km at 10 min after burst. The slab model given in Table 4 is based on Table 3 given in the Greifingers' report. In this connection, eigenangles are given in Table 4 rather than the sine of eigenangles given by the Greifingers and Table 4 has been expanded over the original by linearly interpolating the sine of the eigenangle between inputs in Table 3 of the Greifingers' report. The original Griefinger data would allow for a 13-slab model.

Figures 19 and 20 show amplitude and phase comparisons of the principal field components EZ and HY as a function of disturbance motion along the path described by $\alpha = -57.44°$, D = 500 km (see Figure 2). The range is taken to be 4 Mm. In these calculations mmax = 60 and skip = 100 km. The maximum EZ field fade is about 3.5 dB, and the maximum HY field fade exceeds 4 dB, while phase advances approach 40°. Jaggyness due to slab variation is evident. The bulge in the plots close to S = 3 Mm is indicative of partial sum convergence problems. This could be smoothed out by choosing a larger skip value (say 200

Table 4. Eigenangles and slab radii used to model nuclear environment associated with 2 Mt of fission products at 1000 km altitude 10 min after burst time (from Greifinger & Greifinger, 1977).

SLAB-N	ρ(N)-Km	Θr(Deg)	Θi(Deg)	SLAB-N	ρ(N)-Km	Θr(Deg)	Θi(Deg)
1	400	72.288	-51.263	24	3050	77.090	-43.718
2	475	72.356	-51.085	25	3200	77.504	-43.396
3	550	72.424	-50.906	26	3250	77.978	-42.701
4	625	72.493	-50.726	27	3300	78.468	-41.995
5	700	72.563	-50.545	28	3350	78.976	-41.279
6	775	72.759	-50.308	29	3400	79.501	-40.553
7	850	72.957	-50.071	30	3425	79.746	-40.098
8	925	73.157	-49.832	31	3450	79.996	-39.637
9	1000	73.359	-49.593	32	3475	80.251	-39.172
10	1125	73.192	-49.204	33	3500	80.512	-38.700
11	1250	73.020	-48.813	34	3525	80.832	-38.411
12	1375	72.845	-48.419	35	3550	81.156	-38.120
13	1500	72.667	-48.024	36	3575	81.487	-37.830
14	1625	73.274	-47.591	37	3600	81.823	-37.540
15	1750	73.895	-47.160	38	3650	81.842	-37.121
16	1875	74.529	-46.731	39	3700	81.861	-36.696
17	2000	75.178	-46.305	40	3750	81.880	-36.266
18	2150	75.351	-45.906	41	3800	81.897	-35.831
19	2300	75.528	-45.502	42	3950	82.057	-34.910
20	2450	75.707	-45.095	43	4100	82.222	-33.961
21	2600	75.889	-44.683	44	4250	82.391	-32.981
22	2750	76.283	-44.361	45	4400	82.565	-31.969
23	2900	76.683	-44.039	46	∞	82.346	-32.993

km). It could in principle also be smoothed out by choosing a larger mmax, although experience has shown that convergence with increasing mmax is very slow.

Figures 21 and 22 shows EZ and HY amplitude and phase behavior for a disturbance path described by $\alpha = -57.44°$, D = 3500 km (see Figure 2). The range is 4 Mm so that the center of the disturbance passes within about 420 km of the receiver. Again mmax = 60 and skip = 100 km. The kink in the HY amplitude plot and the break in the derivatives of the phase plots in the neighborhood of S = -3 Mm is again due to inadequacy of the partial sum convergence. Jaggyness due to slab variation is also evident.

Figures 23 through 26 show amplitude and phase plots as a function of range for the nuclear depressed environment. For Figures 23 and 24 the disturbance is centered on the transmitter-receiver path at a distance of 500 km from the transmitter. For Figures 25 and 26 the disturbance is also centered on the transmitter-receiver path but at a distance of 3500 km from the transmitter. In both cases, mmax = 60 and skip = 100 km. Amplitude variations fall between about 1.5 and -5 dB and phase changes between about -10° and 40°.

Considered as a final illustration of the programs application is a polar cap boundary model based on the 22 November 1982 weak solar proton event (SPE), whose effects on ELF propagation were examined by ray trace methods by Field, Warber, and Joiner [1986]. Following the latter reference, the diffuse polar cap boundary is assumed to extend between about 50° and 55° latitude, and to be 0.5 to 1 Mm in width, and the sine of the eigenangle is taken to be:

$$S = S_{SPE} + (S_{AMB} - S_{SPE})/\left[1 + \exp\{-7.3[r-0.5(r_1 + r_2)]/\Delta r\}\right], \tag{41}$$

where

$$S_{SPE} = 1.25 - 0.117i = \text{sine of eigenangle in disturbed polar cap} \tag{42}$$

$$S_{AMB} = 1.15 - 0.063i = \text{sine of eigenangle in night ambient} \tag{43}$$

$$r_1 = 3 \text{ Mm} \tag{44}$$

$$r_2 = r_1 + \Delta r \tag{45}$$

$$\Delta r = \text{transition width} \tag{46}$$

and r is measured from the north geomagnetic pole.

Figures 27 though 30 show amplitude and phase results for the EZ and HY components for a hypothetical motion of the pole along a path described by α = 90°, D = 1802.78 km (see Figure 2). The calculations have been performed with mmax = 60, skip = 50 km, Δr = 1 Mm, and 33 slabs. The range is 3 Mm in Figures 27 and 28 and 4 Mm in Figures 29 and 30. For the parameters of Figures 27 through 30, grazing incidence on the polar cap boundary corresponds (the case of interest to Field, et al. [1986]) to s $\simeq$ 3 Mm. It will be seen that the HY field fade there is about 1.2 dB. This corresponds favorably with the ray trace estimate of 1.4 dB given by Field, et al. [1986]. However, the phase advance of about 25° to 30° indicated in Figures 28 and 30 for the grazing incident geometry appears to be inconsistent with measurements which indicate little change from ambient.

Figures 31 and 32 show amplitude and phase results, respectively, for the EZ and HY field components as a function of range. Again the calculations have been performed with mmax = 60, skip = 100 km, Δr = 1 Mm, and 33 slabs. The center location of the disturbance is given by x0 = 1802.78 km, y0 = 3000 km. This corresponds to grazing incidence on the polar cap boundary.

Irregularities in the curves in the neighborhood of the range $\approx$ 3.6 Mm is indicative of partial sum convergence problems, which could be smoothed out with a larger value of the input parameter, skip.

Figures 33 through 36 are analogues of Figures 27 through 30, the only difference being that Δr = 0.5 Mm instead of 1 Mm. It will be seen that the field fades for s $\simeq$ 3 Mm, the grazing incidence geometry, is $\lesssim$ 1 dB, and the phase advance is somewhat in excess of 25°. This is to be contrasted with the ray theory results of Field, et al., which suggest 6-dB fades for the polar cap transition thickness Δr = 500 km.

Figures 37 and 38 are the analogues of Figures 31 and 32. They have been plotted for mmax = 60, skip = 100 km, number of slabs = 33, and Δr = 0.5 Mm. Again the evidence would point to fades considerably less than the 6 dB suggested by ray trace arguments. Again, irregularities in the neighborhood of 3.6 Mm indicate partial sum convergence problems. Other wiggles are due to slab-to-slab variations.

Figures 38 and 40 show amplitude and phase results as a function of range for a path which penetrates the polar cap. The center of the disturbance is described by x0 = 3354.10 km, y0 = 1000 km. Thus it is a path whose closest point of approach to the geomagnetic North Pole is 1000 km. Though a slight standing wave pattern will be seen, the amplitude falloff rate is about 0.75 dB/Mm, which is consistent with the difference between the disturbed and ambient attenuation rates.

REFERENCES

Ferguson, J.A., L.R. Hitney and R.A. Pappert, A program to compute vertical electric ELF fields in a laterally inhomogeneous earth ionosphere waveguide, NOSC TR 851, prepared for the Def. Nucl. Agency, 62 pp, Dec 1982.

Field, E.C., C.R. Warber and R.G. Joiner, "Focusing and shadowing of ELF signals," Radio Sci., 21 (3), 511-517, 1986.

Field, E.C. and R.G. Joiner, "An integral equation approach to long wave propagation in a non stratified earth ionosphere waveguide," Radio Sci., 14 (6), 1057-1068, 1979.

Field, E.C. and R.G. Joiner, "Effects of lateral ionospheric gradients on ELF propagation," Radio Sci., 17 (3), 693-700, 1982.

Greifinger, C. and P. Greifinger, Effects of a cylindrically symmetric disturbance on ELF propagation in the earth ionosphere waveguide, Def. Nucl. Agency. Rep. DNA 4399T, 40 pp. R&D Associates, Marina del Rey, CA, June 1977.

Morfitt, D.G. and C.H. Shellman, 'MODESRCH', an improved computer program for obtaining ELF/VLF/LF mode constants in an earth ionosphere waveguide, Interim report 77T, prepared for the Def. Nucl. Agency by the Naval Electronics Lab. Cen. (now Naval Ocean System Center), 227 pp, Oct 1976.

Pappert, R.A., "Calculated effects of traveling spordic-E on nocturnal ELF propagation. Comparison with measurement," Radio Sci., 20 (2), 229-246, 1985.

Pappert, R.A., "Effects of a large patch of sporadic-E on nighttime propagation at lower elf," J. Atmos. Terr. Phys., 42 (5), 417-425, 1980.

Pappert, R.A. and W.F. Moler, "A theoretical study of ELF normal mode reflection and absorption produced by nighttime ionospheres," J. Atmos. Terr. Phys., 40 (9), 1031-1045, 1978.

Shellman, C.H., Propagation of ELF waves in the spherical earth-ionosphere waveguide with transverse and longitudinal variation of properties, NOSC TD 607, prepared for the Def. Nucl. Agency, 194 pp. June 1983.

Wait, J.R., "On phase changes in very low-frequency propagation induced by an ionospheric depression of finite extent," J. Geophys. Res., 59 (3), 441-445, 1964.

Watson, G.H., A treatise on the theory of Bessel functions, Cambridge University Press, England, 1952, p. 361.

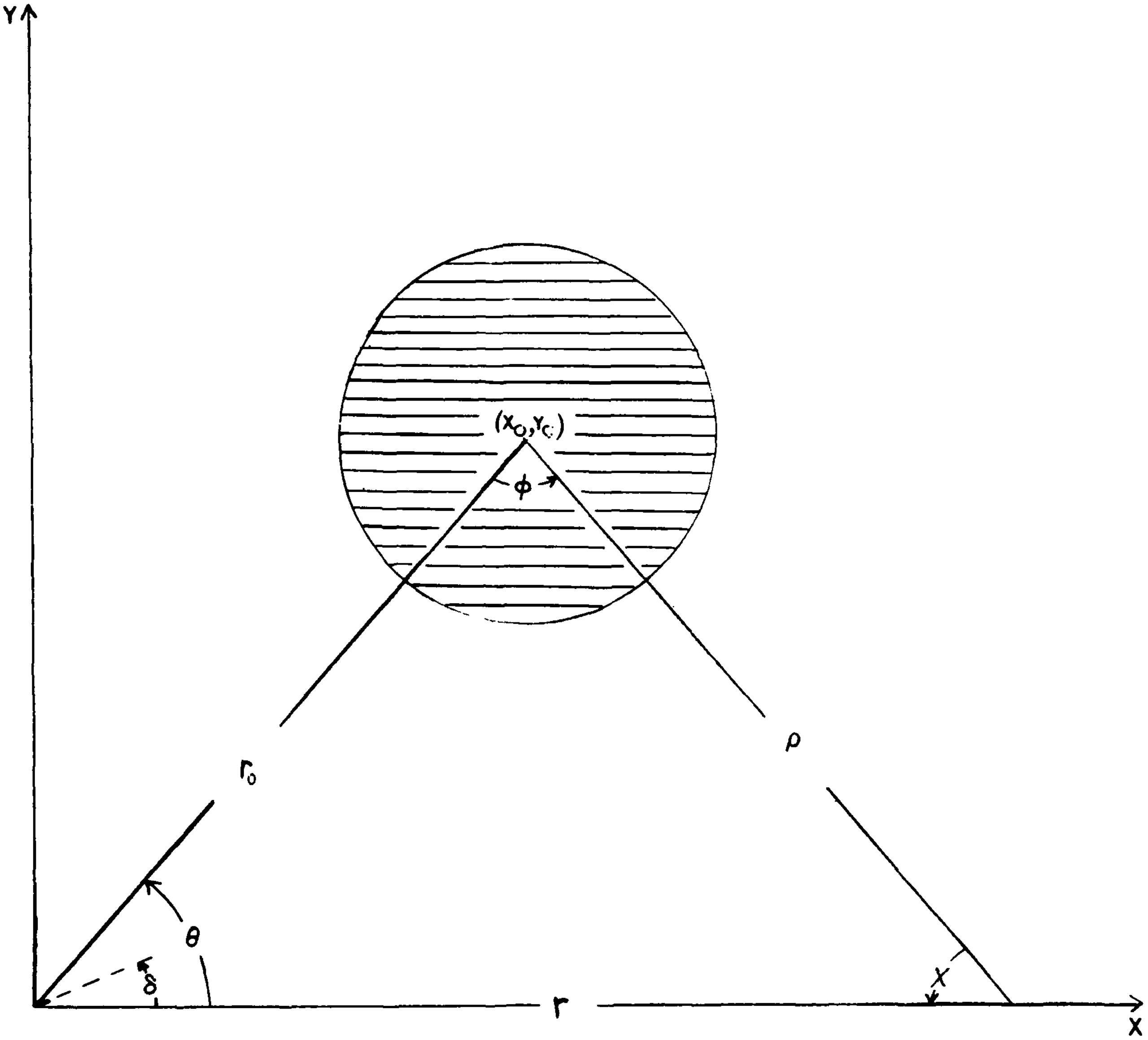


Figure 1. Scattering geometry.

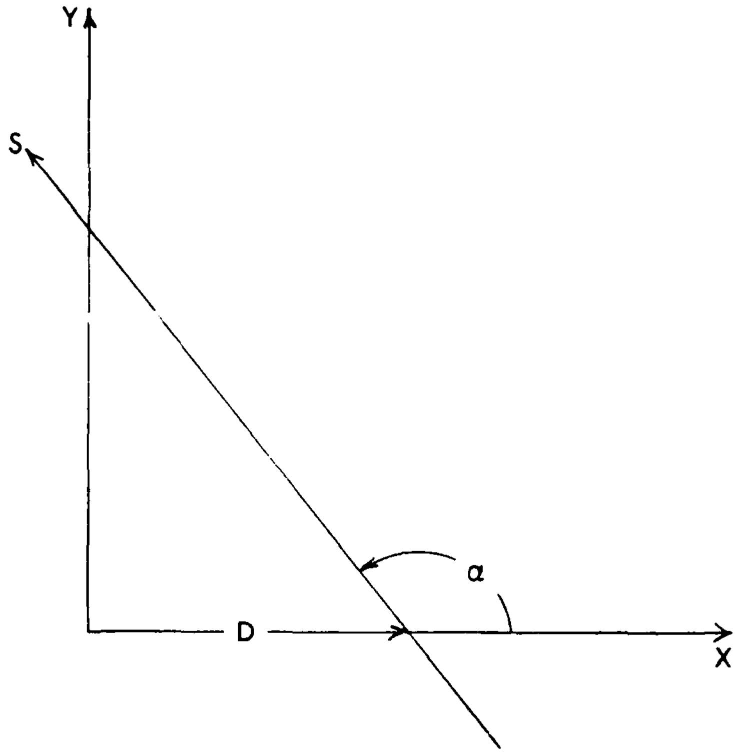


Figure 2. Schematic of disturbance path.

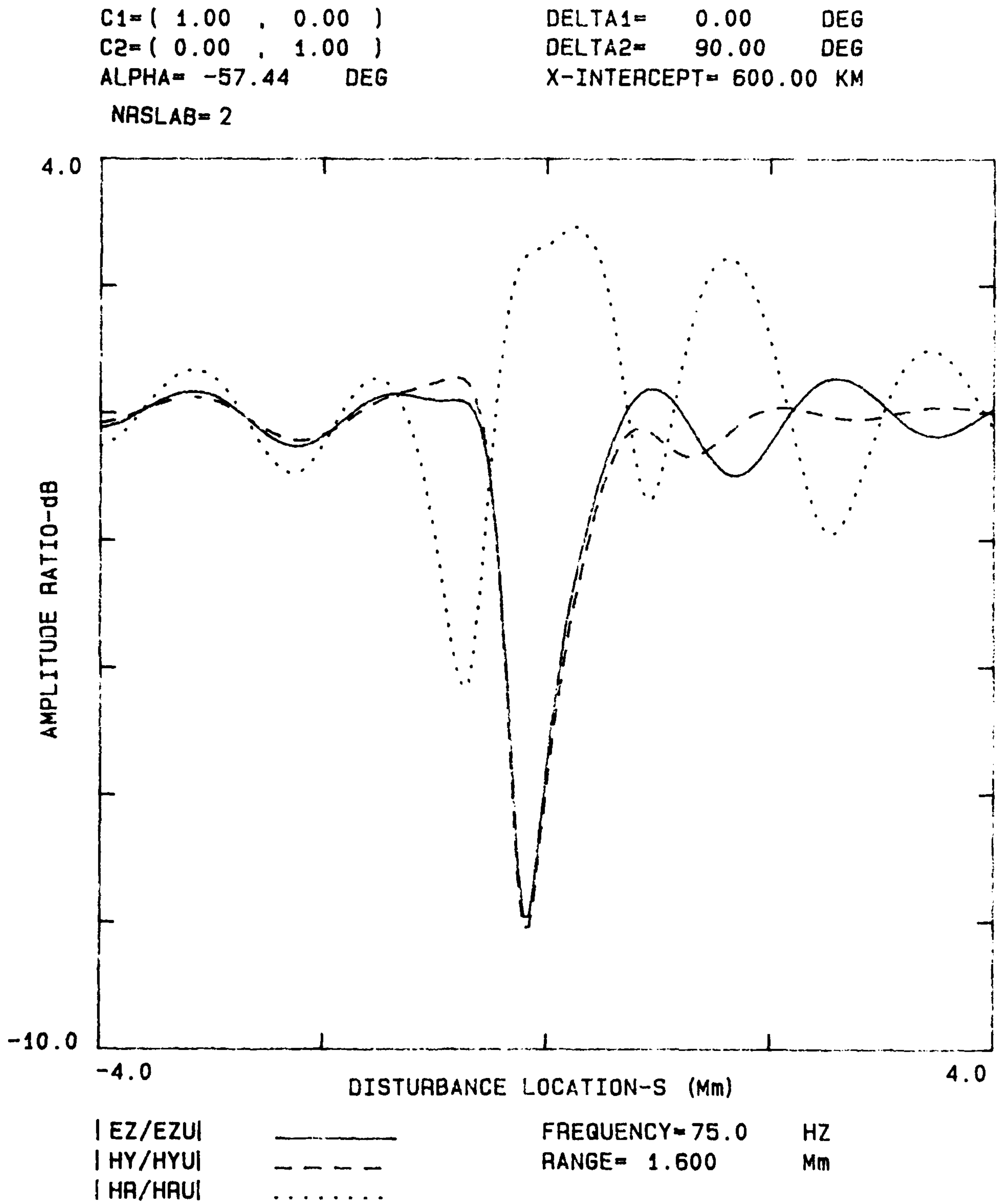


Figure 3. Amplitude ratios vs. disturbance location for homogeneous sporadic-E patch of 500 km radius.

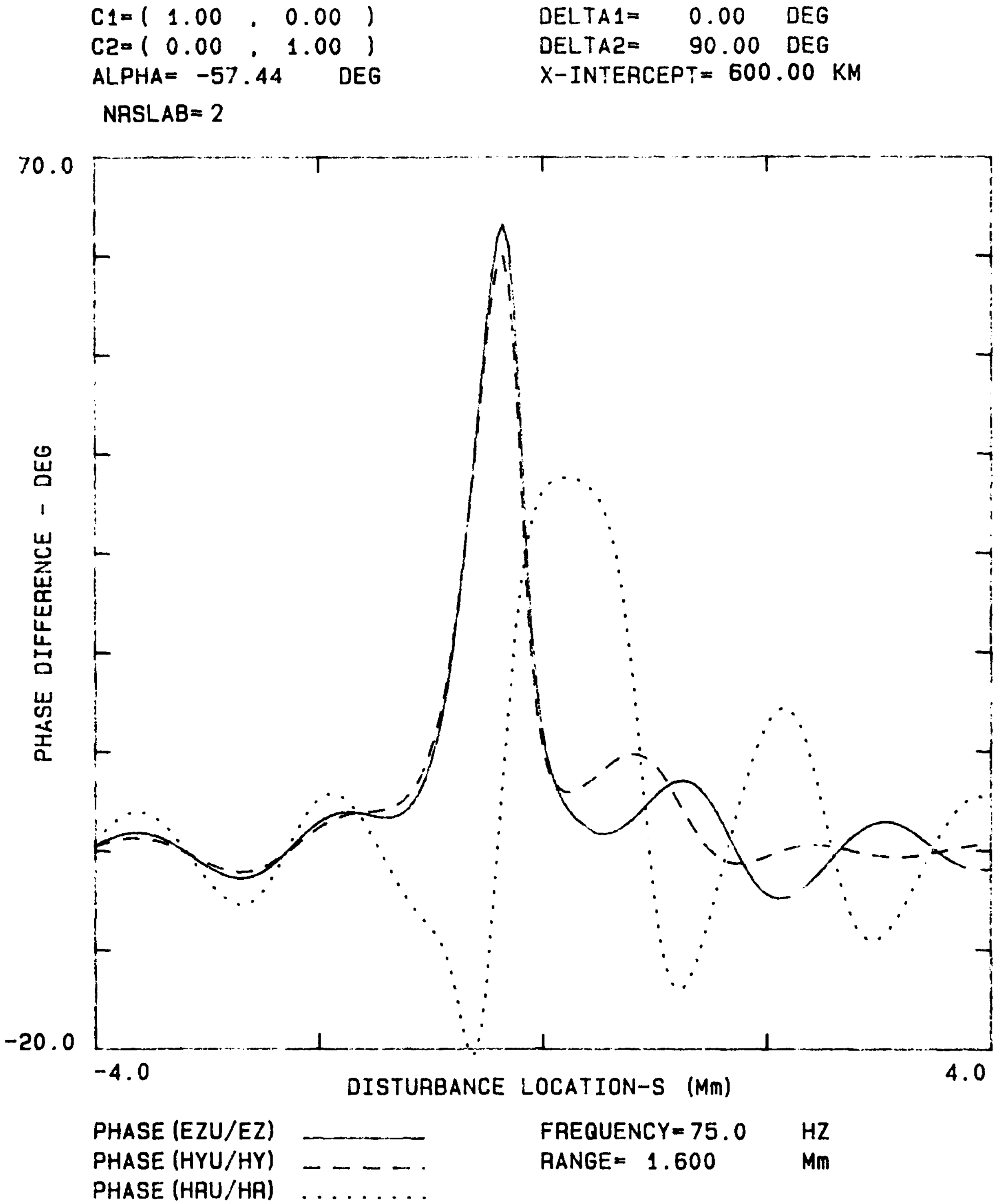


Figure 4. Phase differences vs. disturbance location for homogeneous sporadic-E patch of 500 km radius.

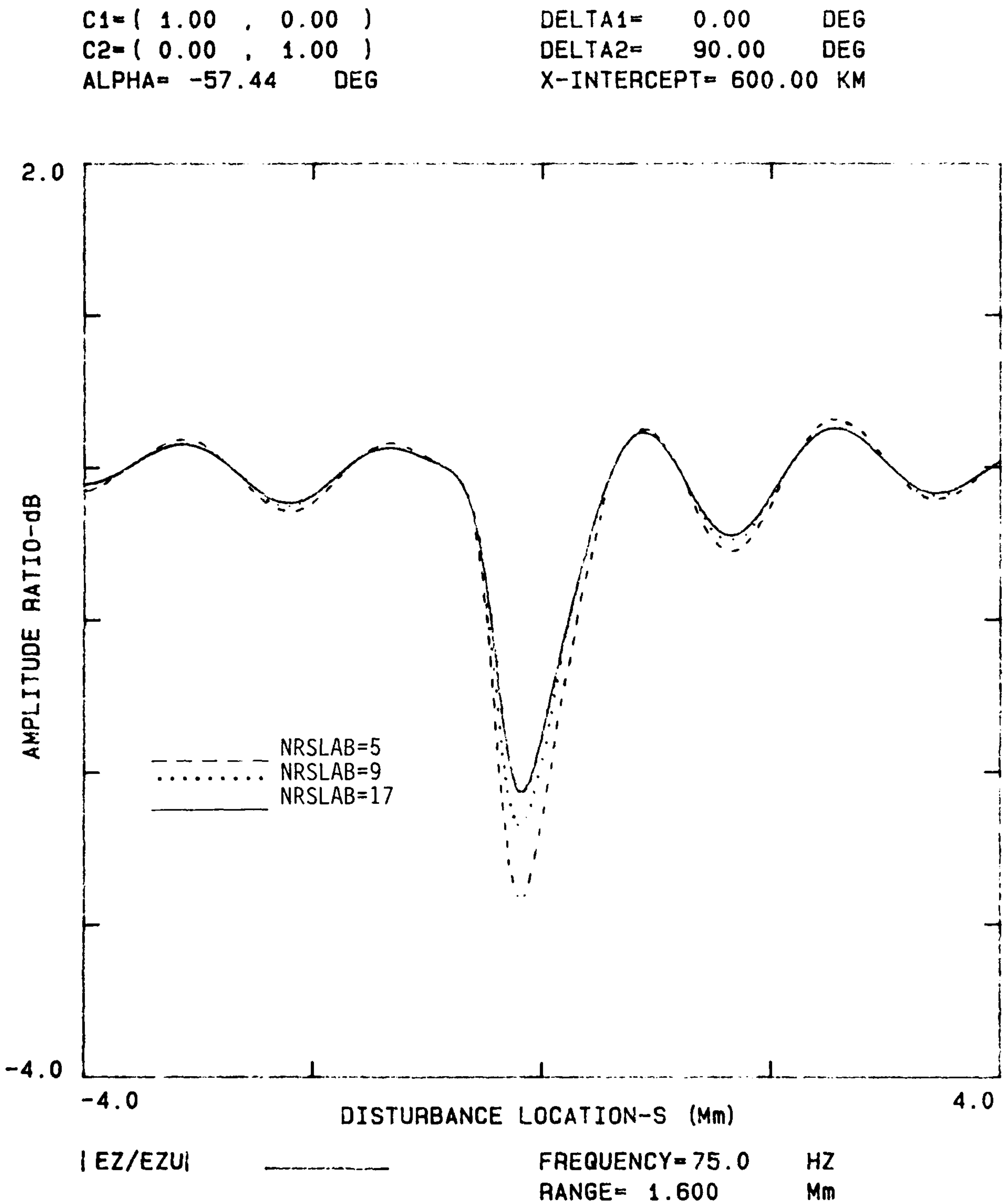


Figure 5. Slab convergence for a linearly inhomogeneous sporadic-E patch of 500 km radius, EZ amplitude ratio.

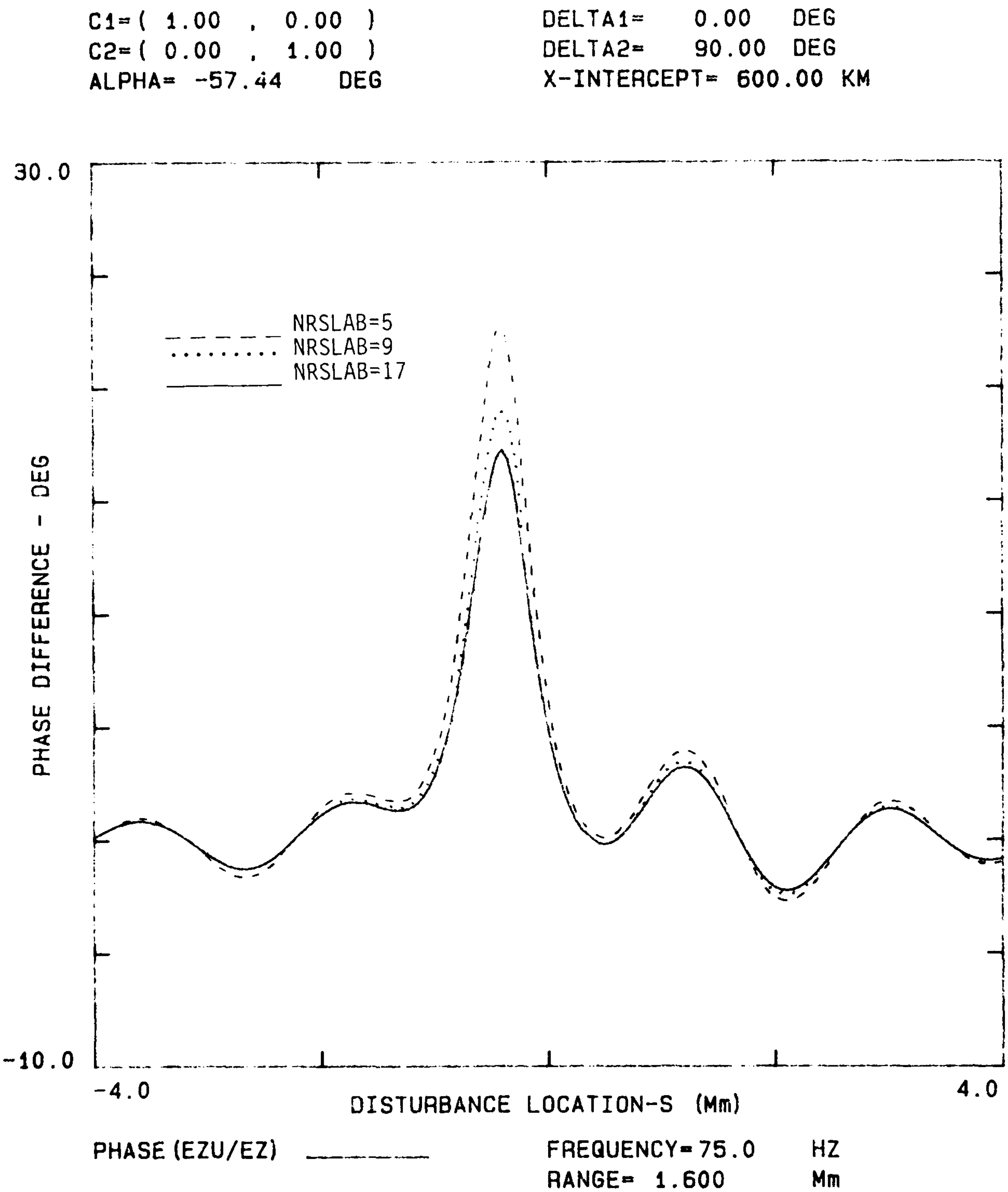


Figure 6. Slab convergence for a linearly inhomogeneous sporadic-E patch of 500 km radius, EZ phase difference.

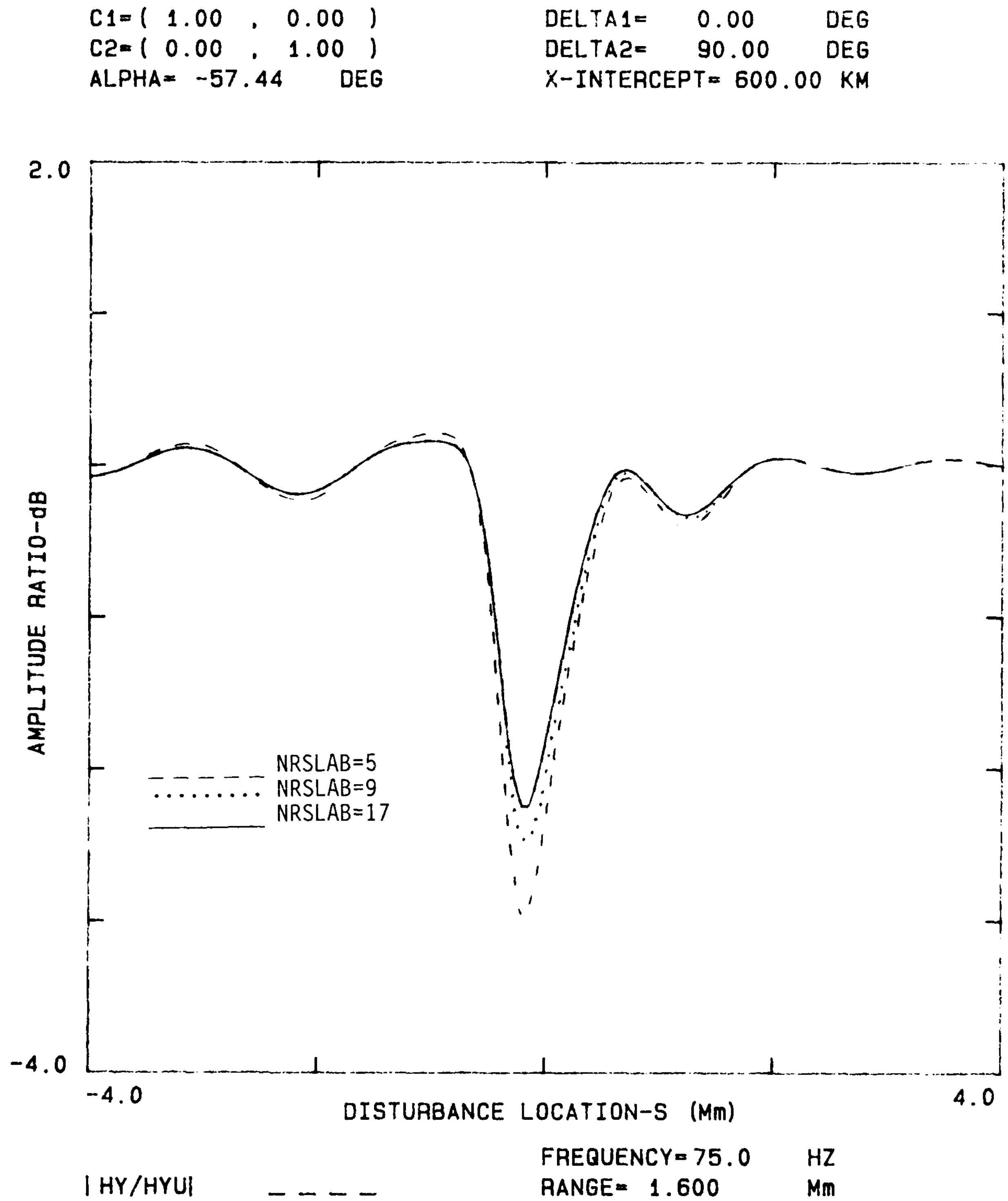


Figure 7. Slab convergence for a linearly inhomogeneous sporadic-E patch of 500 km radius, HY amplitude ratio.

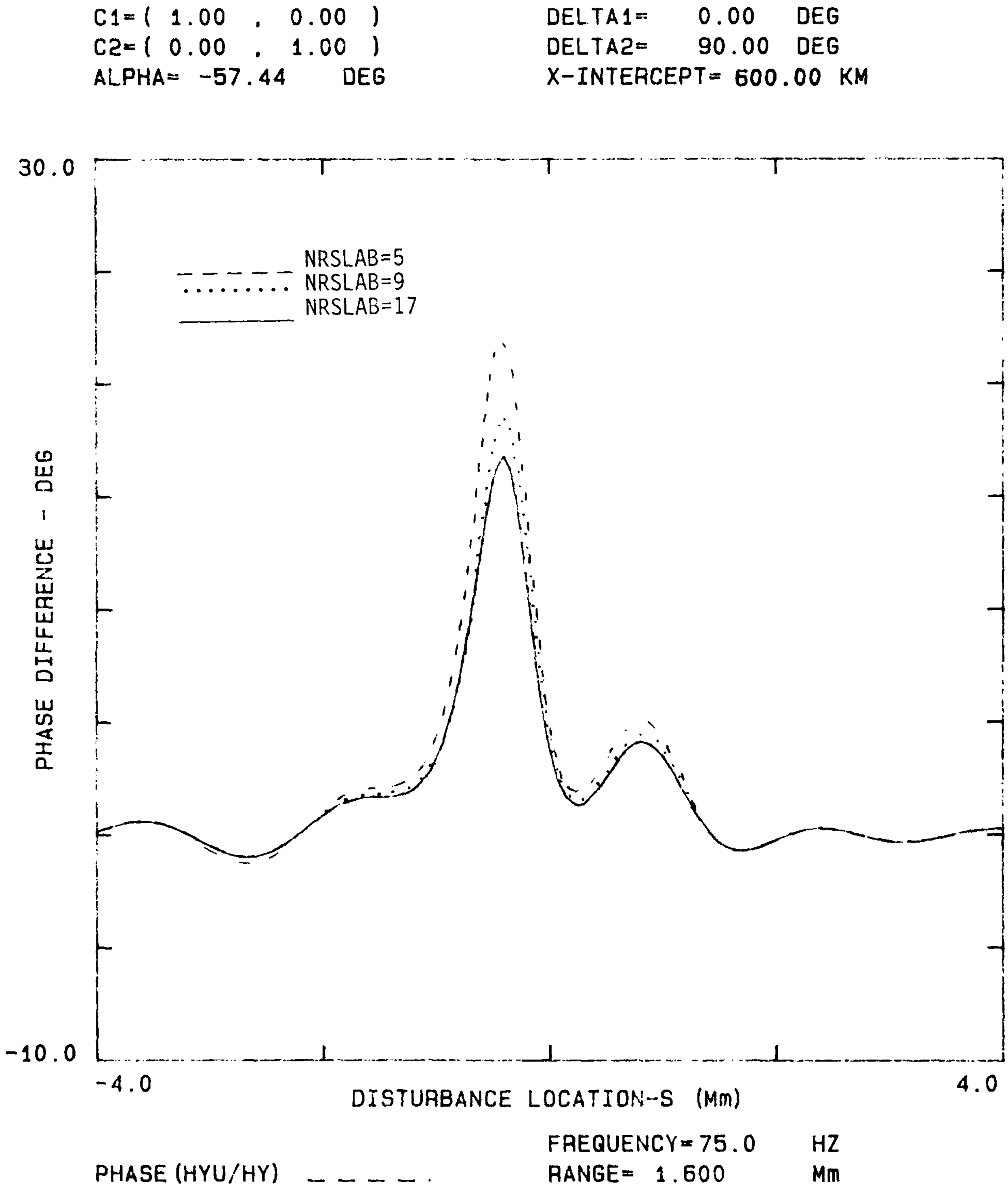


Figure 8. Slab convergence for a linearly inhomogeneous sporadic-E patch of 500 km radius, HY phase difference.

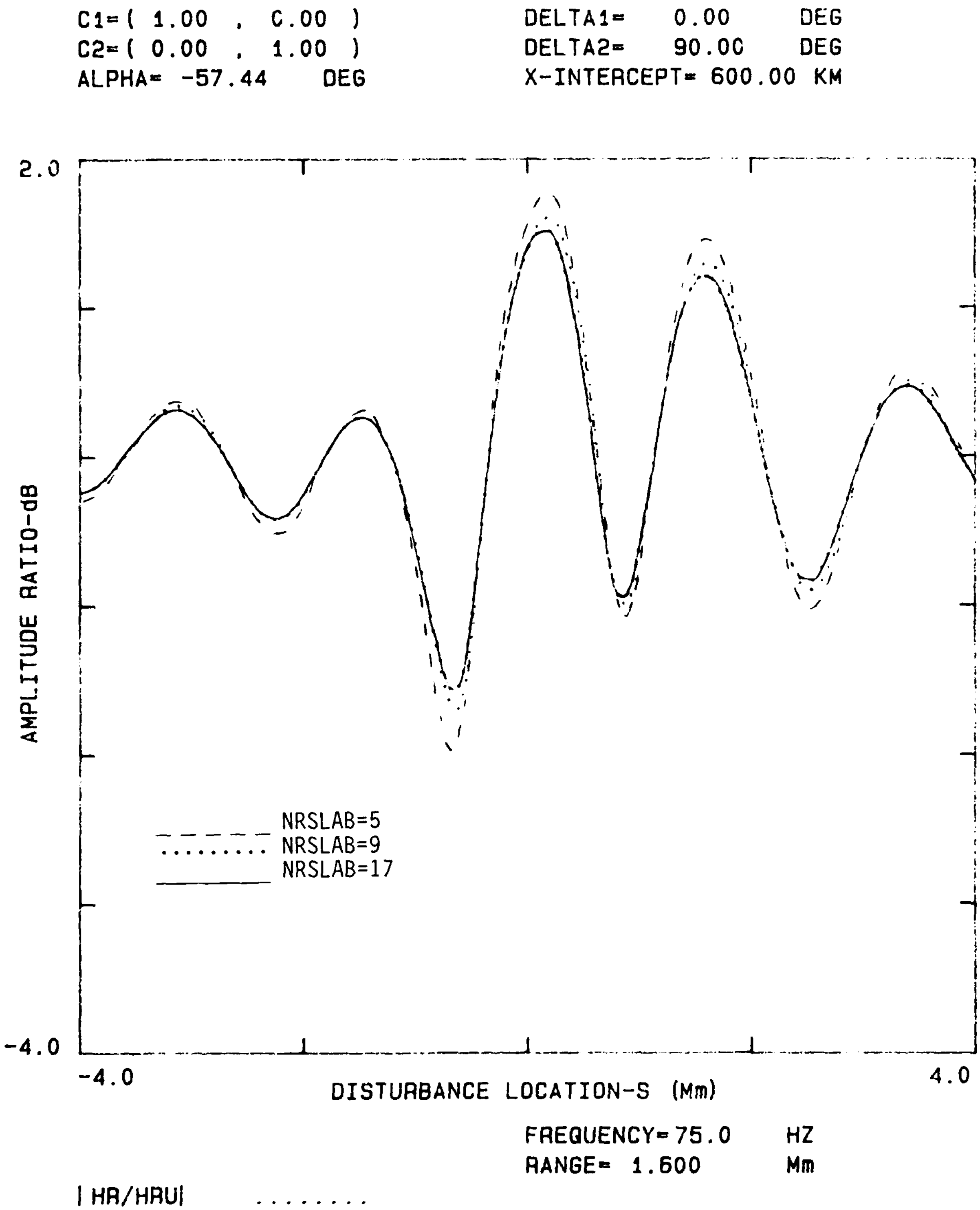


Figure 9. Slab convergence for a linearly inhomogeneous sporadic-E patch of 500 km radius, HR amplitude ratio.

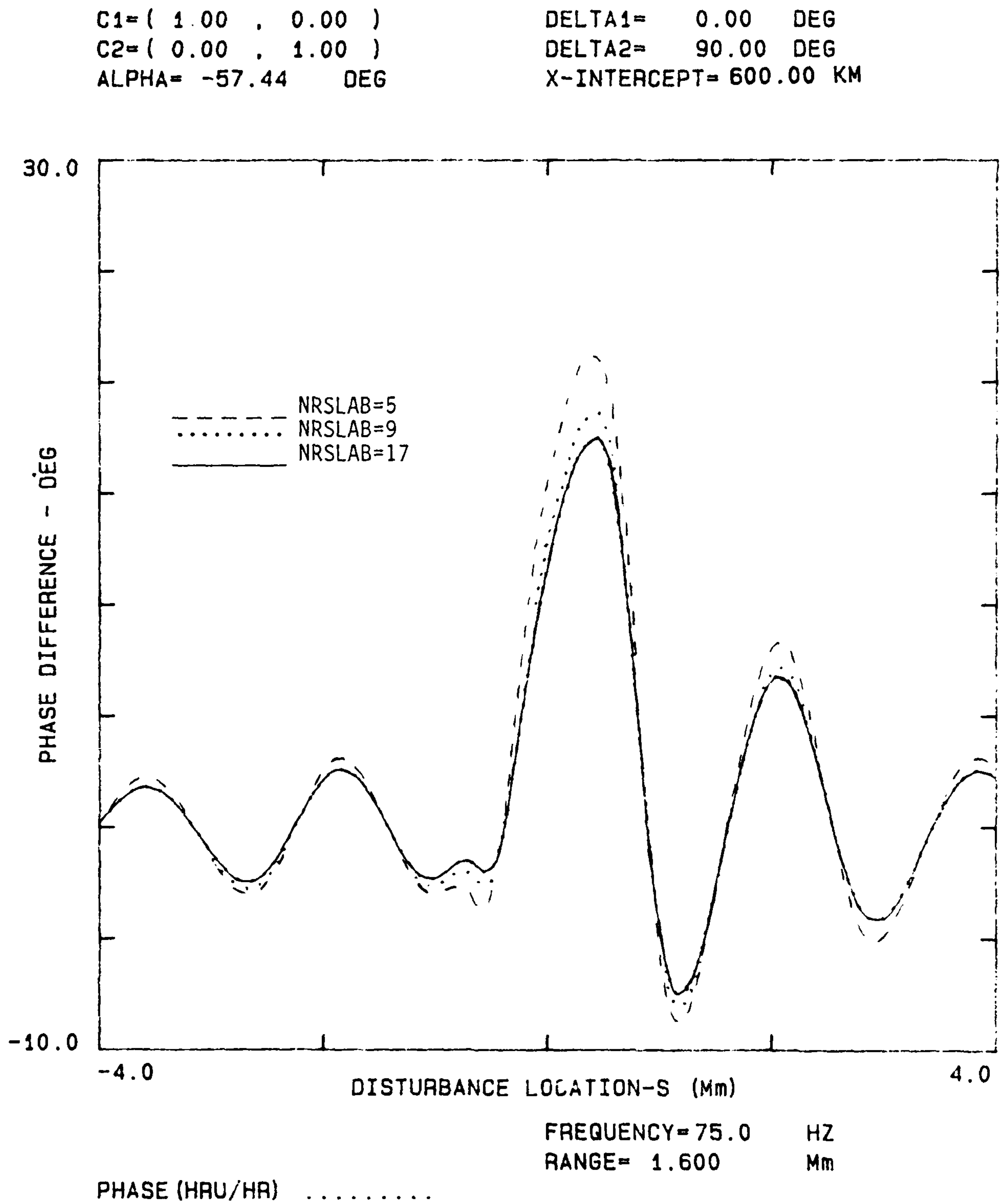


Figure 10. Slab convergence for a linearly inhomogeneous sporadic-E patch of 500 km radius, HR phase difference.

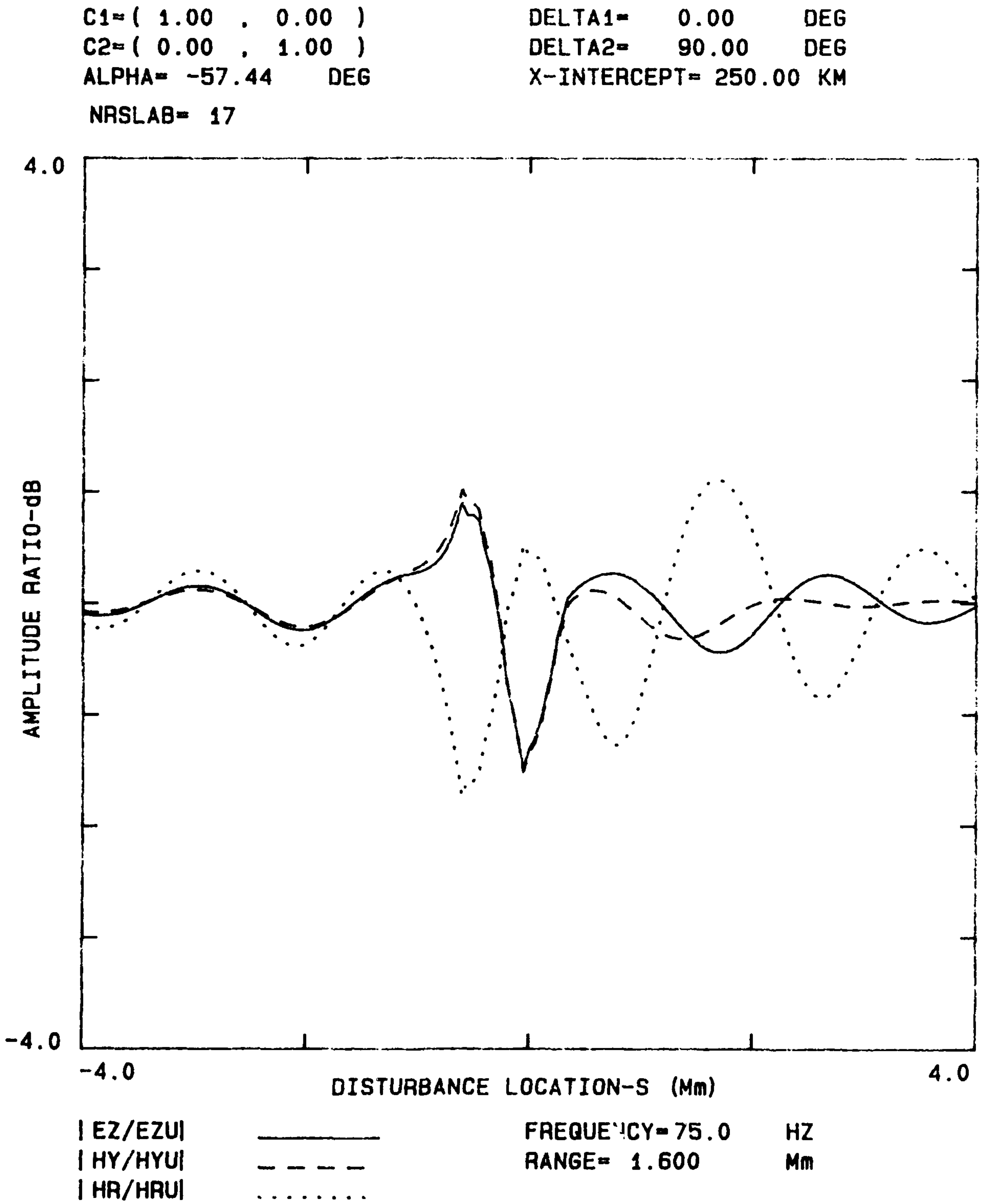


Figure 11. Amplitude ratios vs. disturbance location for a linearly inhomogeneous sporadic-E patch of 500 km radius which passes over the transmitter.

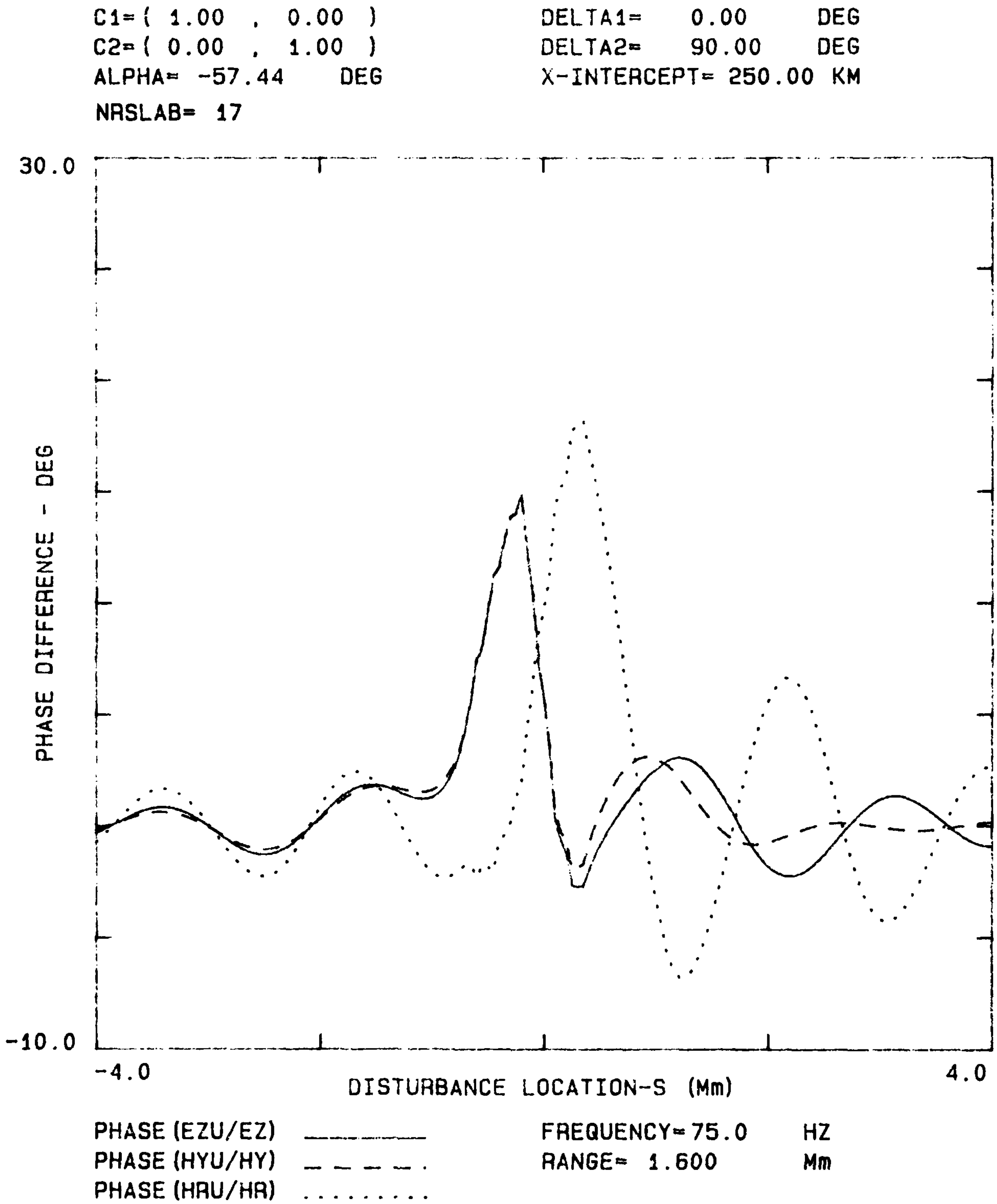


Figure 12. Phase differences vs. disturbance location for a linearly inhomogeneous sporadic-E patch of 500 km radius which passes over the transmitter.

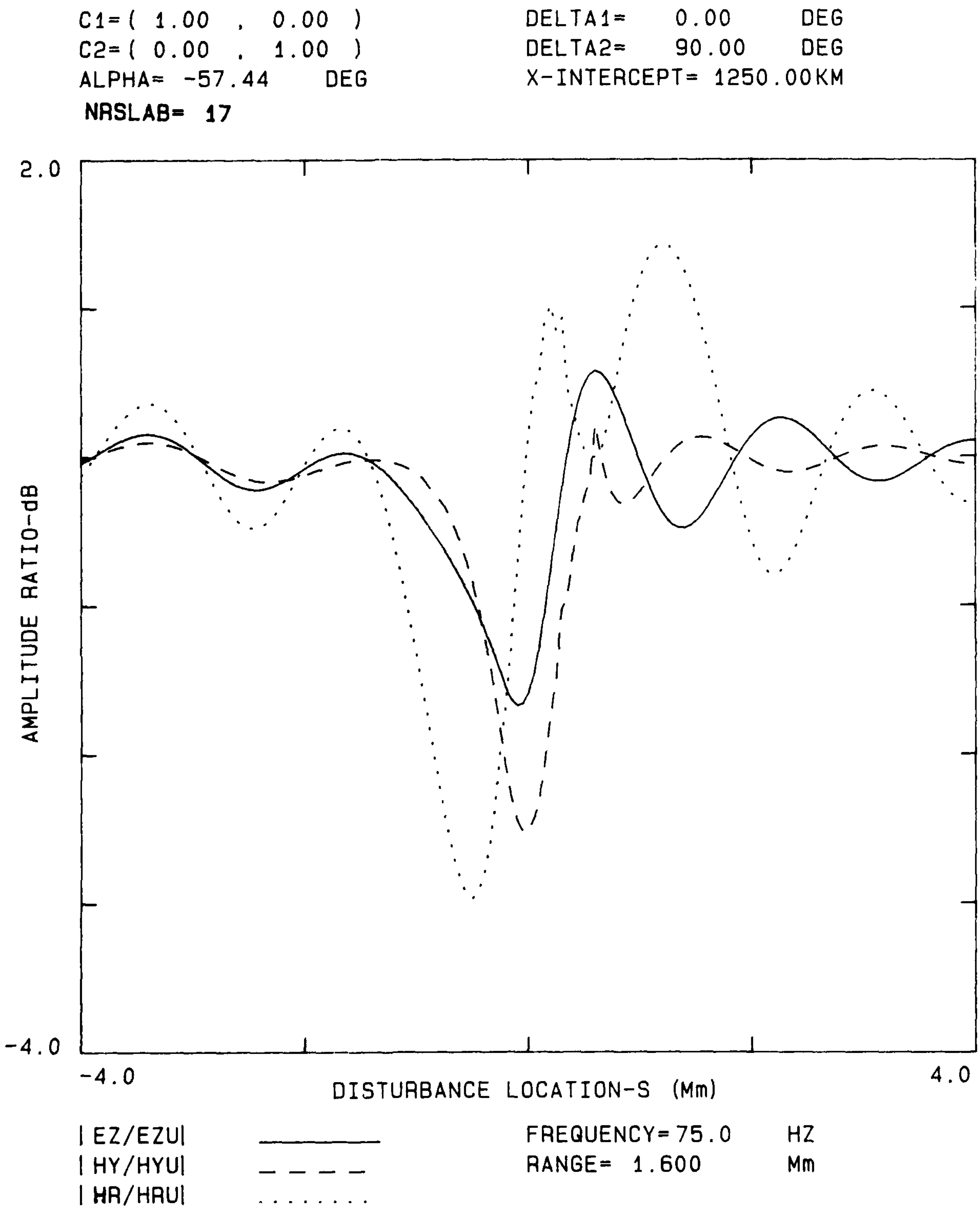


Figure 13. Amplitude ratios vs. disturbance location for a linearly inhomogeneous sporadic-E patch of 500 km radius which passes over the receiver.

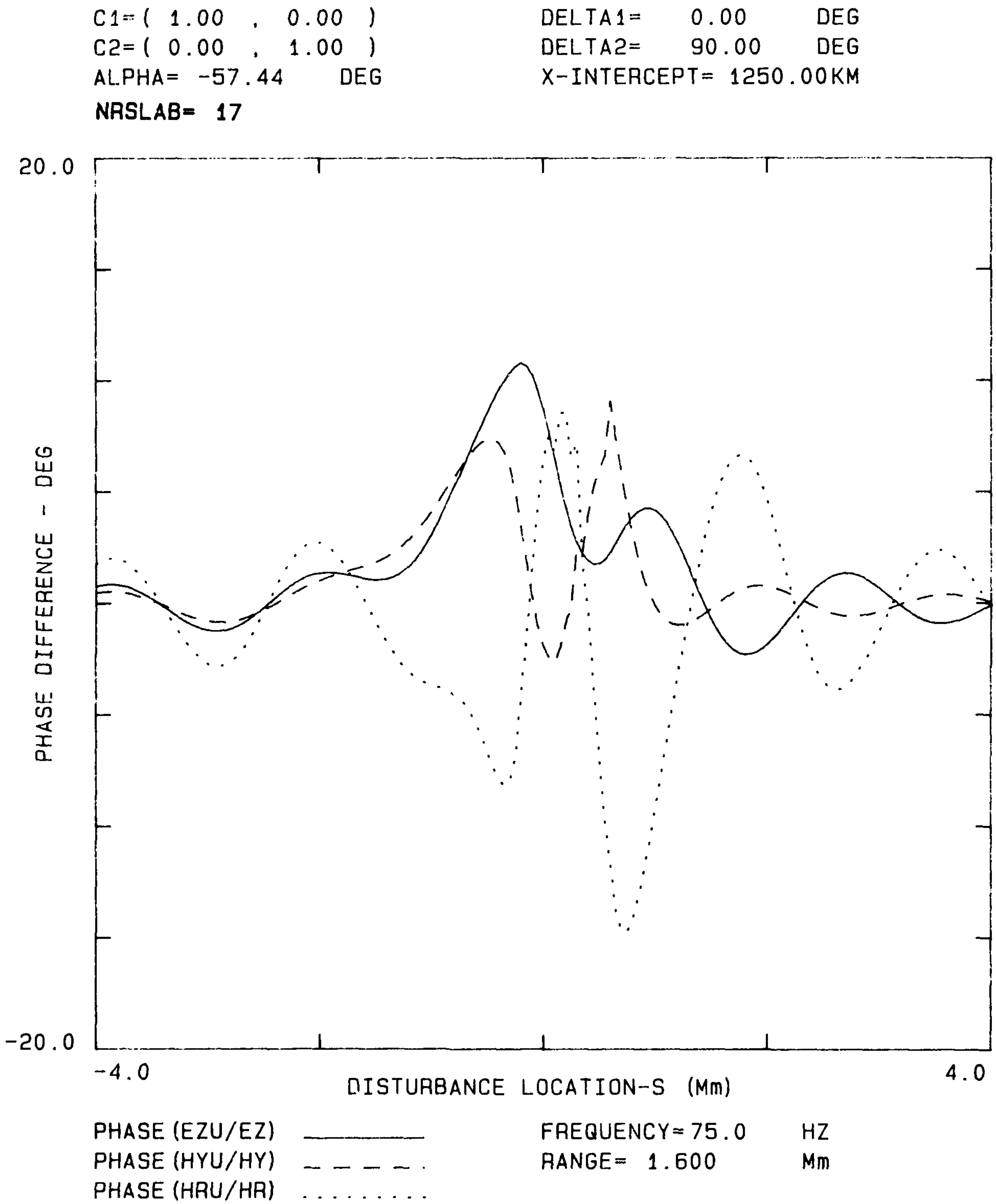


Figure 14. Phase differences vs. disturbance location for a linearly inhomogeneous sporadic-E patch of 500 km radius which passes over the receiver.

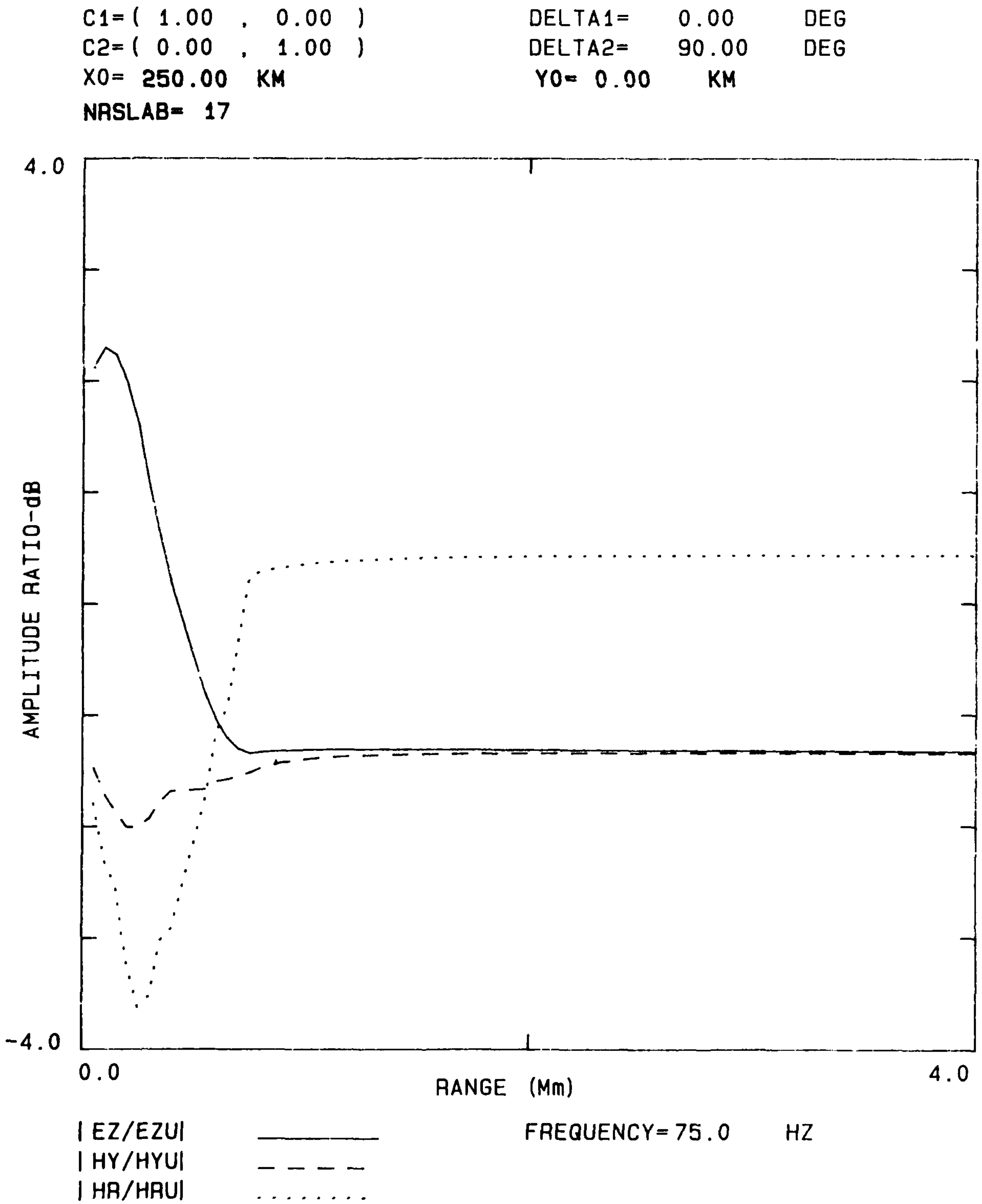


Figure 15. Amplitude ratios vs. range for an on-path linearly inhomogeneous sporadic-E patch of 500 km radius which overlaps the transmitter.

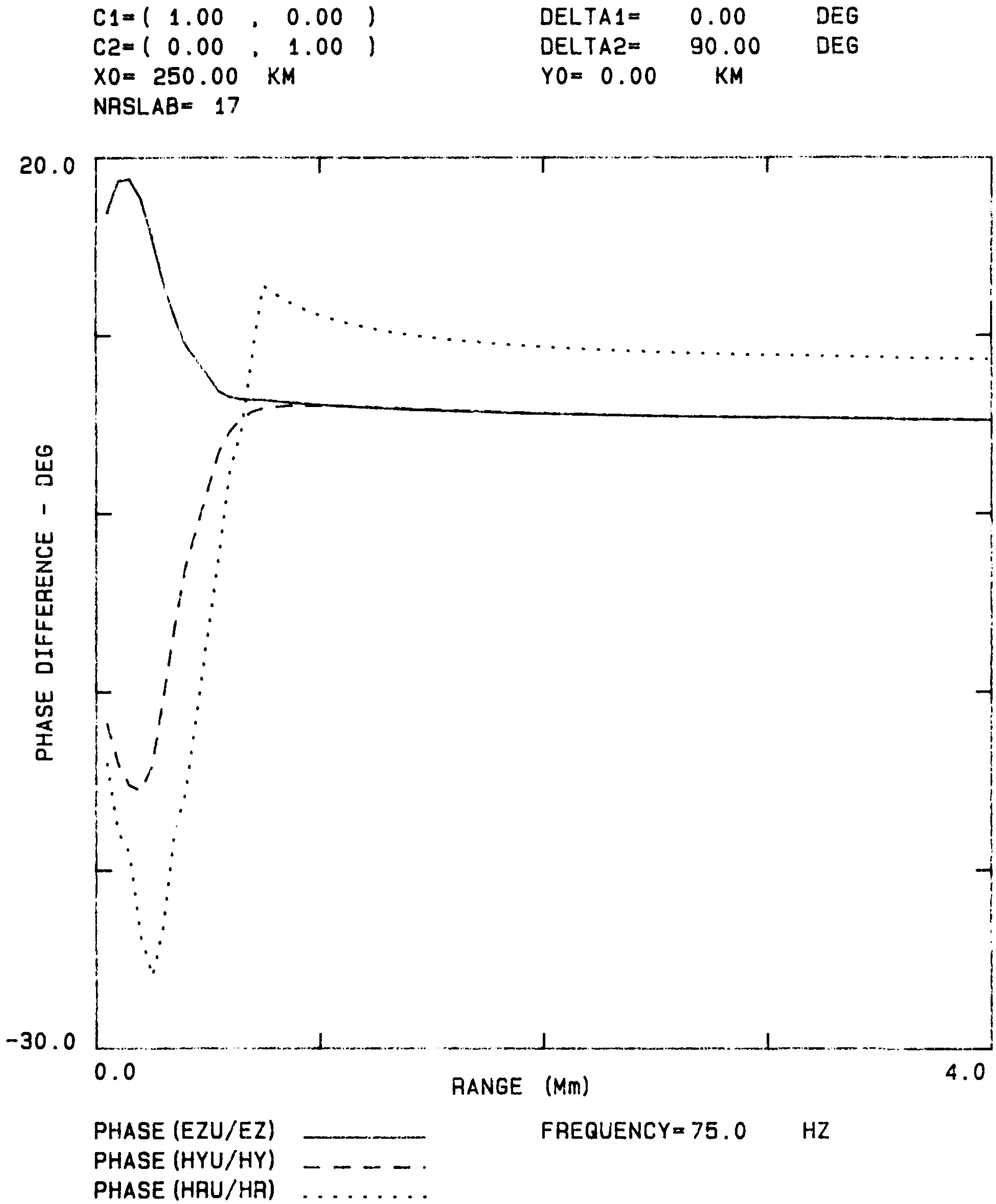


Figure 16. Phase differences vs. range for an on-path linearly inhomogeneous sporadic-E patch of 500 km radius which overlaps the transmitter.

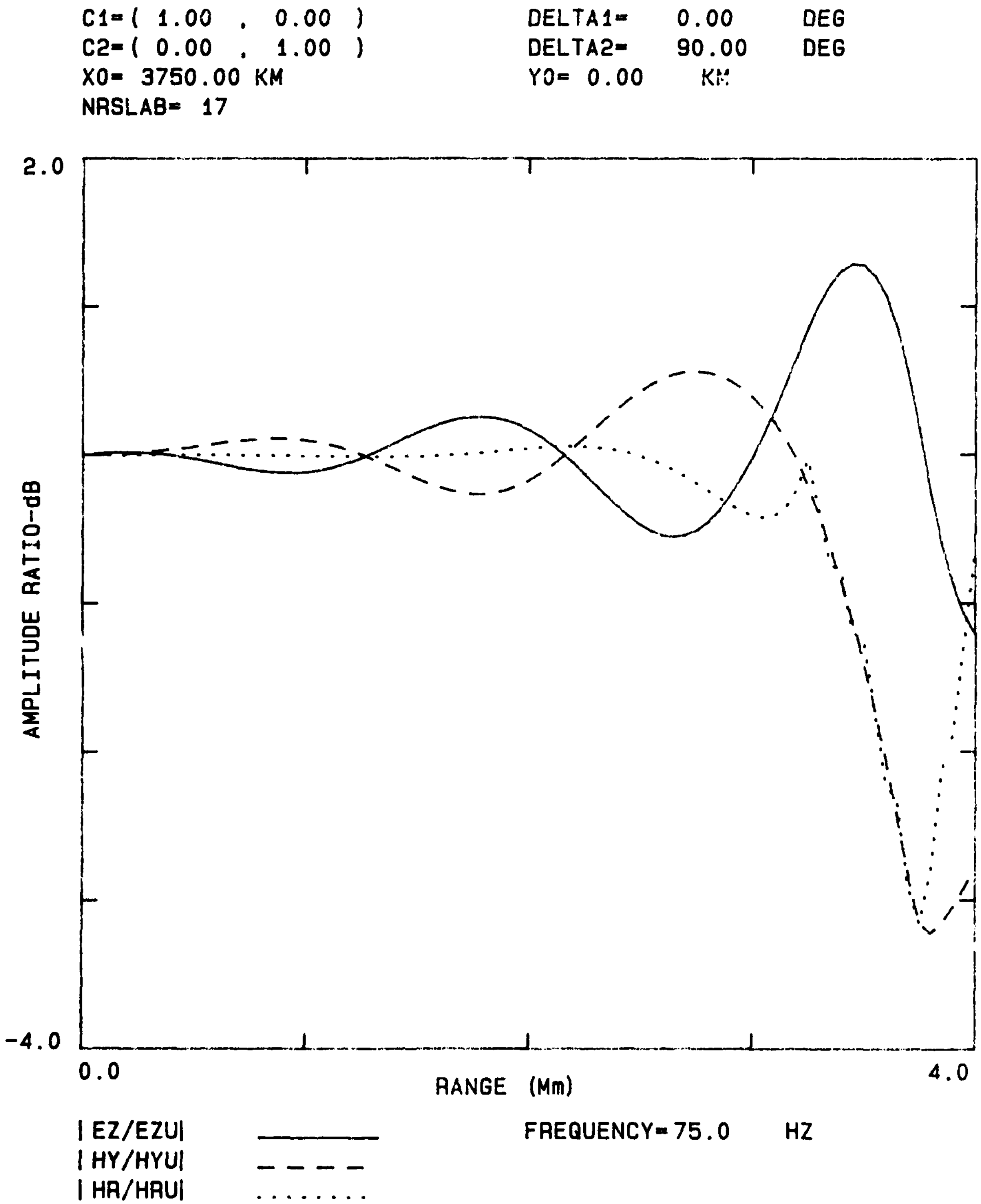


Figure 17. Amplitude ratios vs. range for an on-path linearly inhomogeneous sporadic-E patch of 500 km radius which overlaps the receiver.

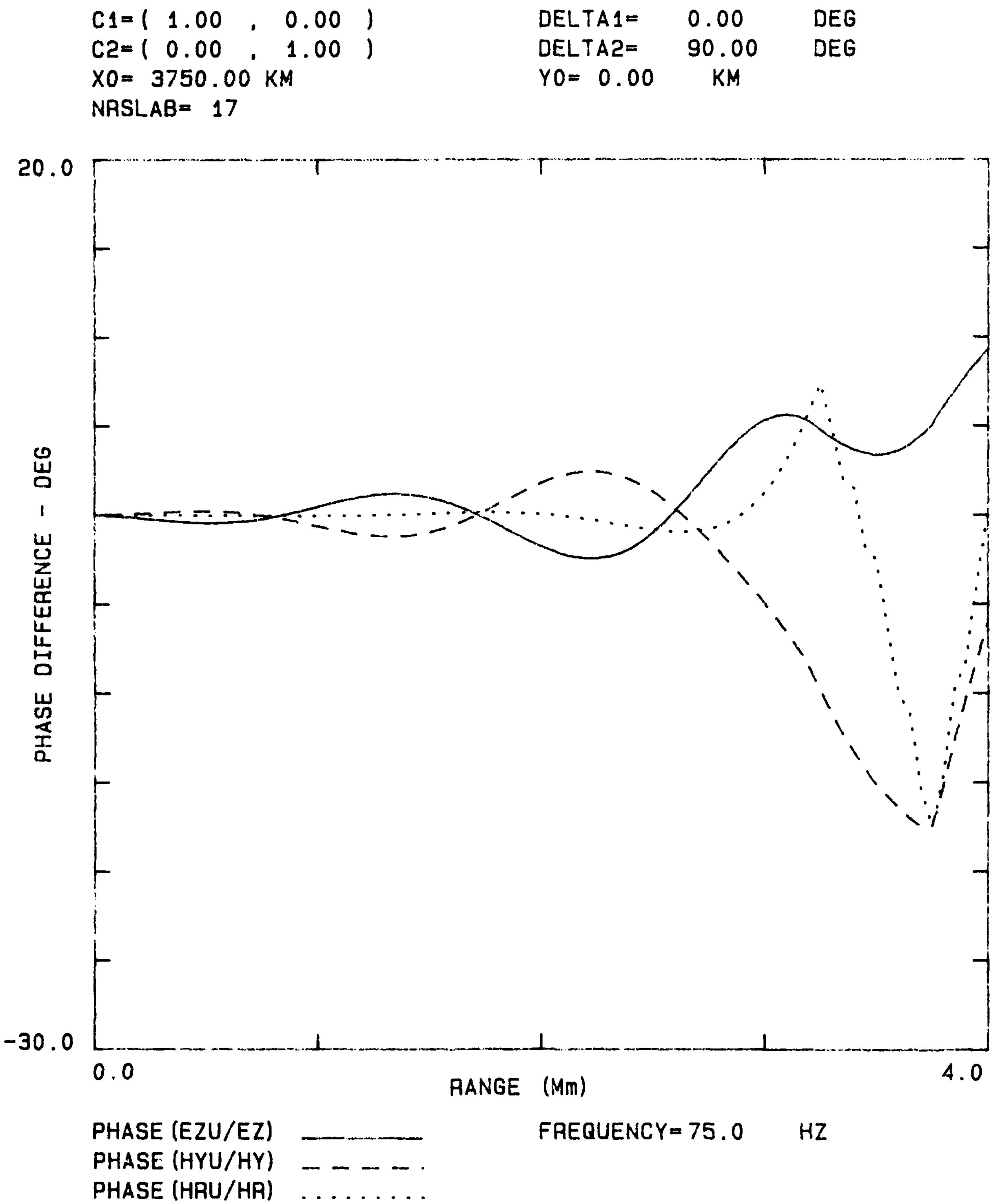


Figure 18. Phase differences vs. range for an on-path linearly inhomogeneous sporadic-E patch of 500 km radius which overlaps the receiver.

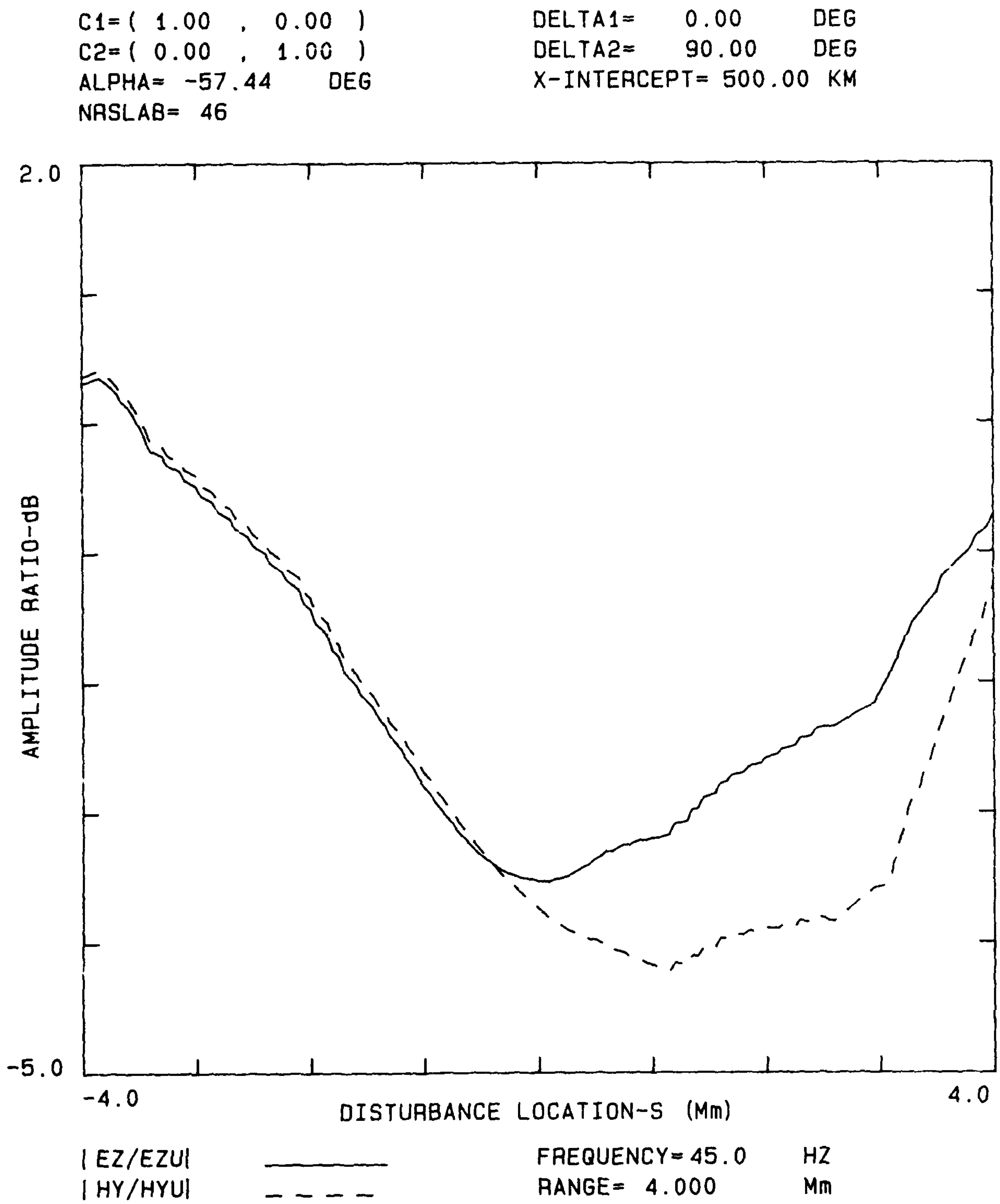


Figure 19. Amplitude ratios vs. disturbance location for a nuclear depression of radius 4.4 Mm. Disturbance center crosses transmitter-receiver path at 500 km.

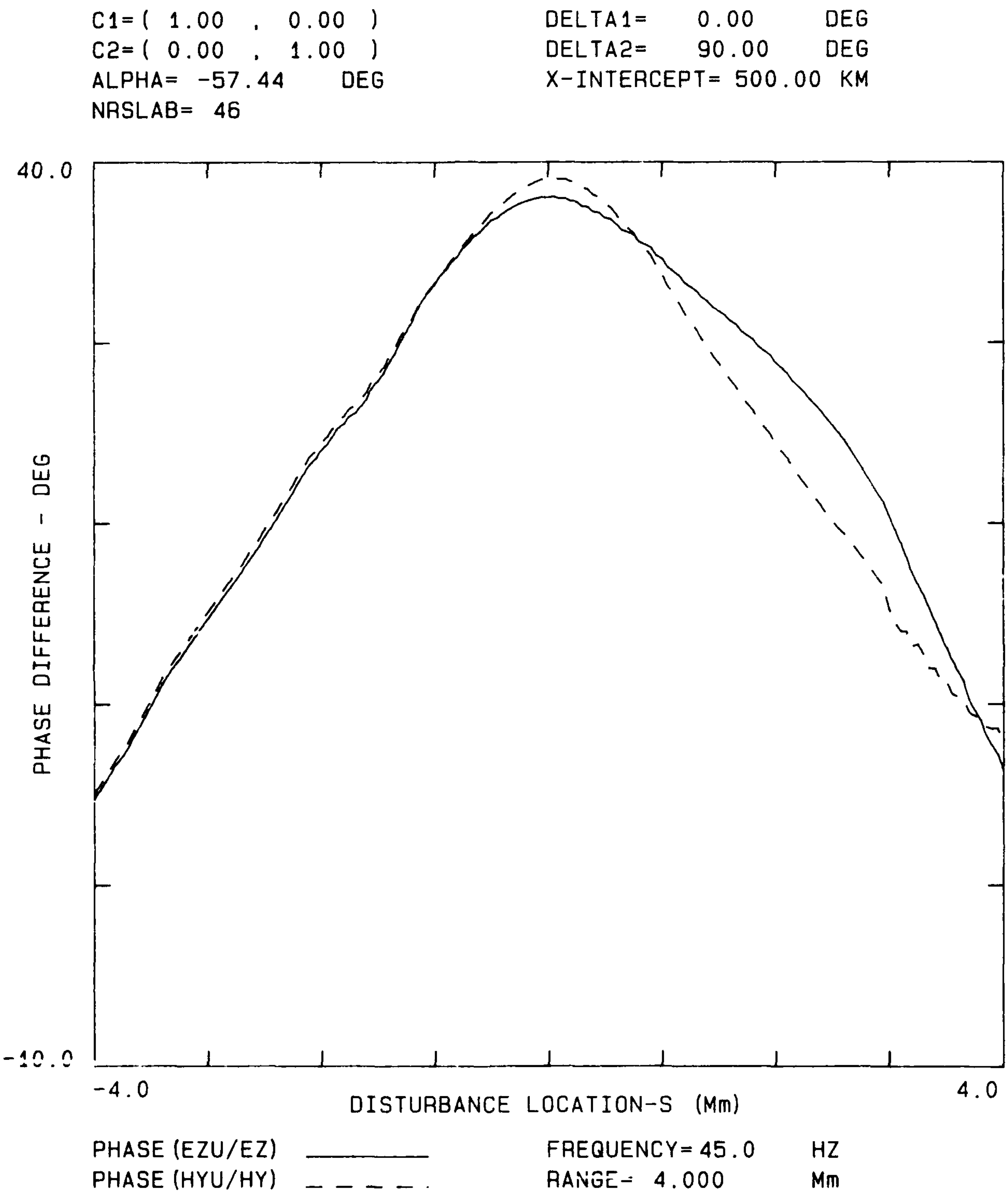


Figure 20. Phase differences vs. disturbance location for a nuclear depression of radius 4.4 Mm. Disturbance center crosses transmitter-receiver path at 500 km.

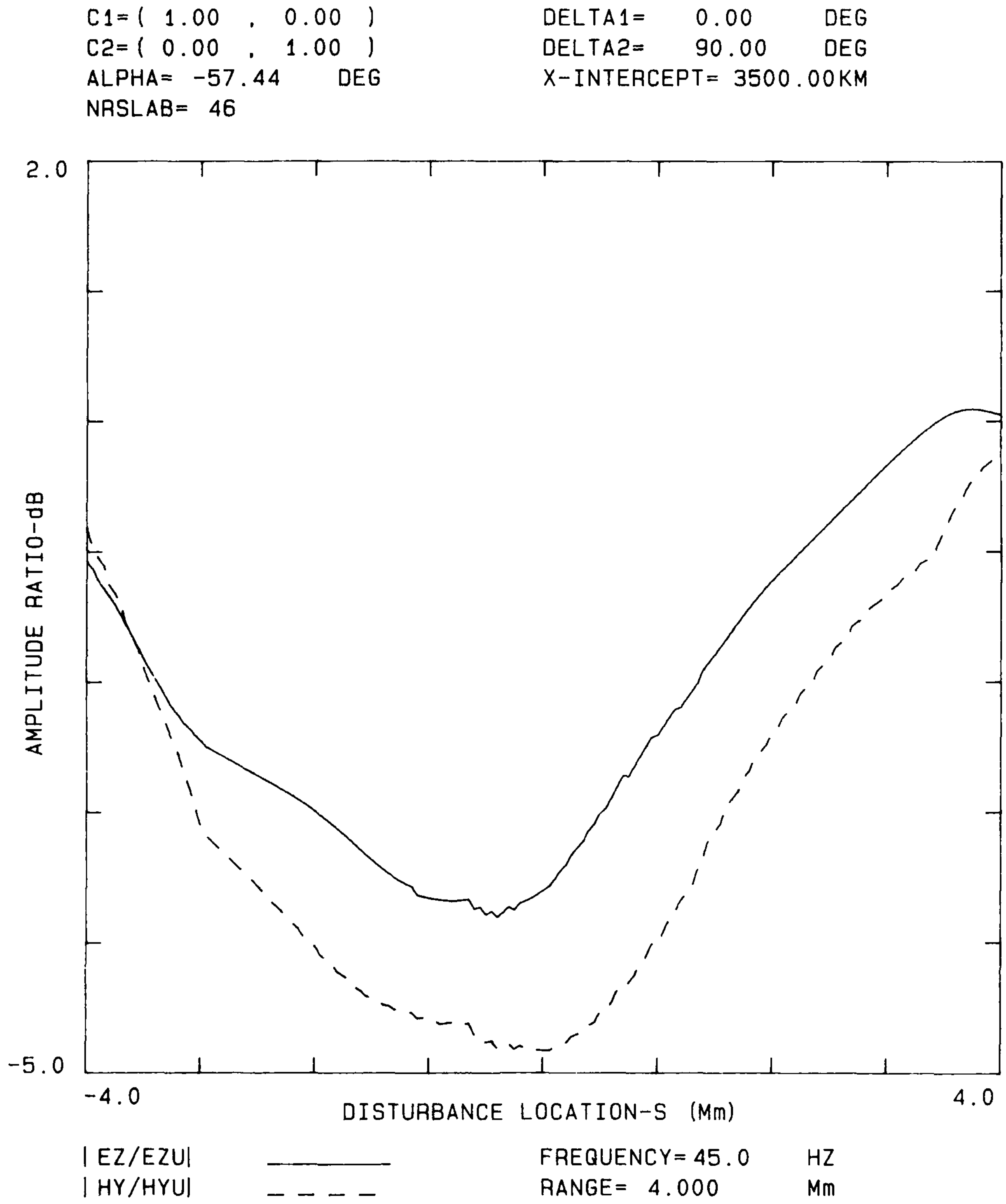


Figure 21. Amplitude ratios vs. disturbance location for a nuclear depression of radius 4.4 Mm. Disturbance center crosses transmitter-receiver path at 3500 km.

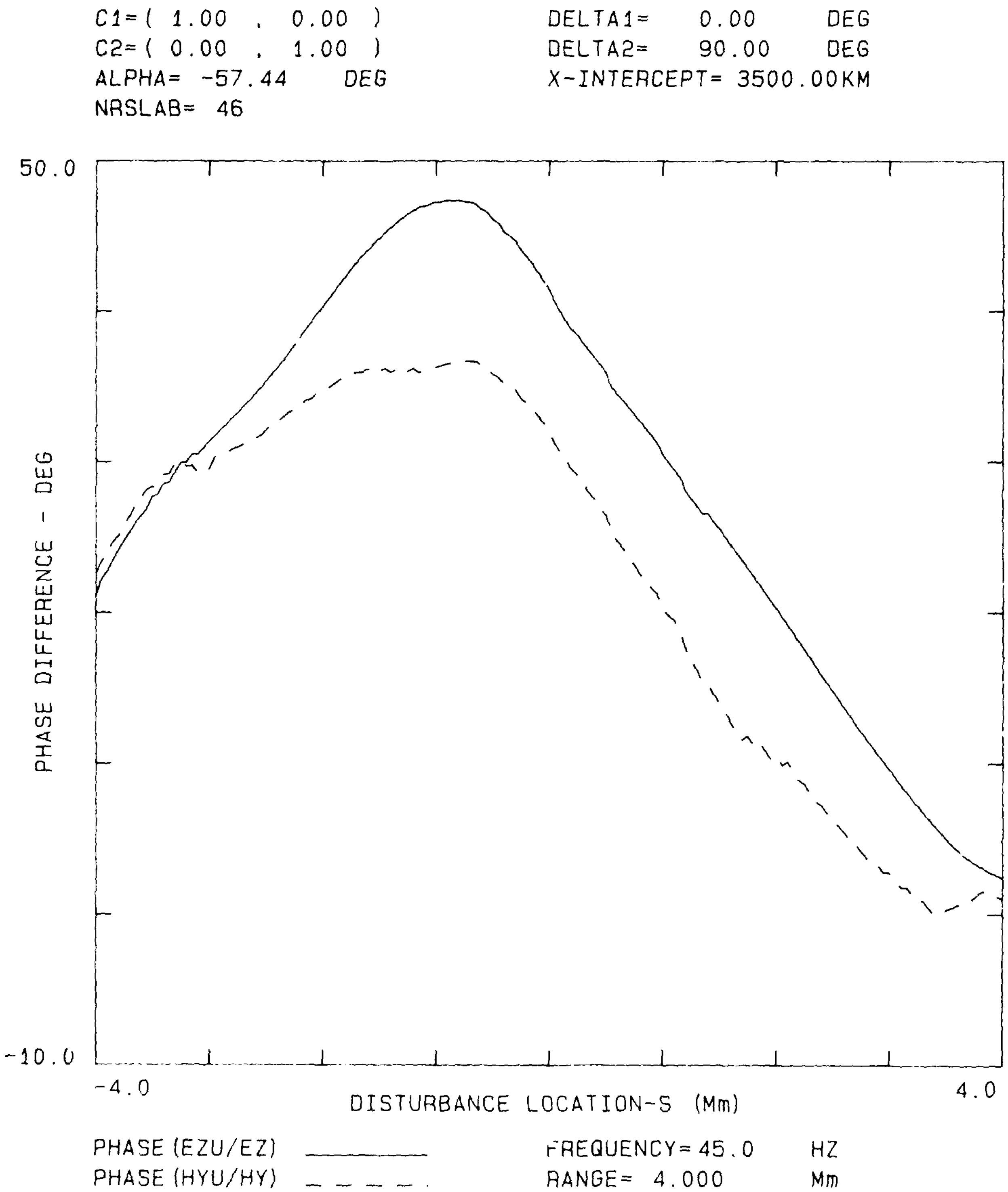


Figure 22. Phase differences vs. disturbance location for a nuclear depression of radius 4.4 Mm. Disturbance center crosses transmitter-receiver path at 3500 km.

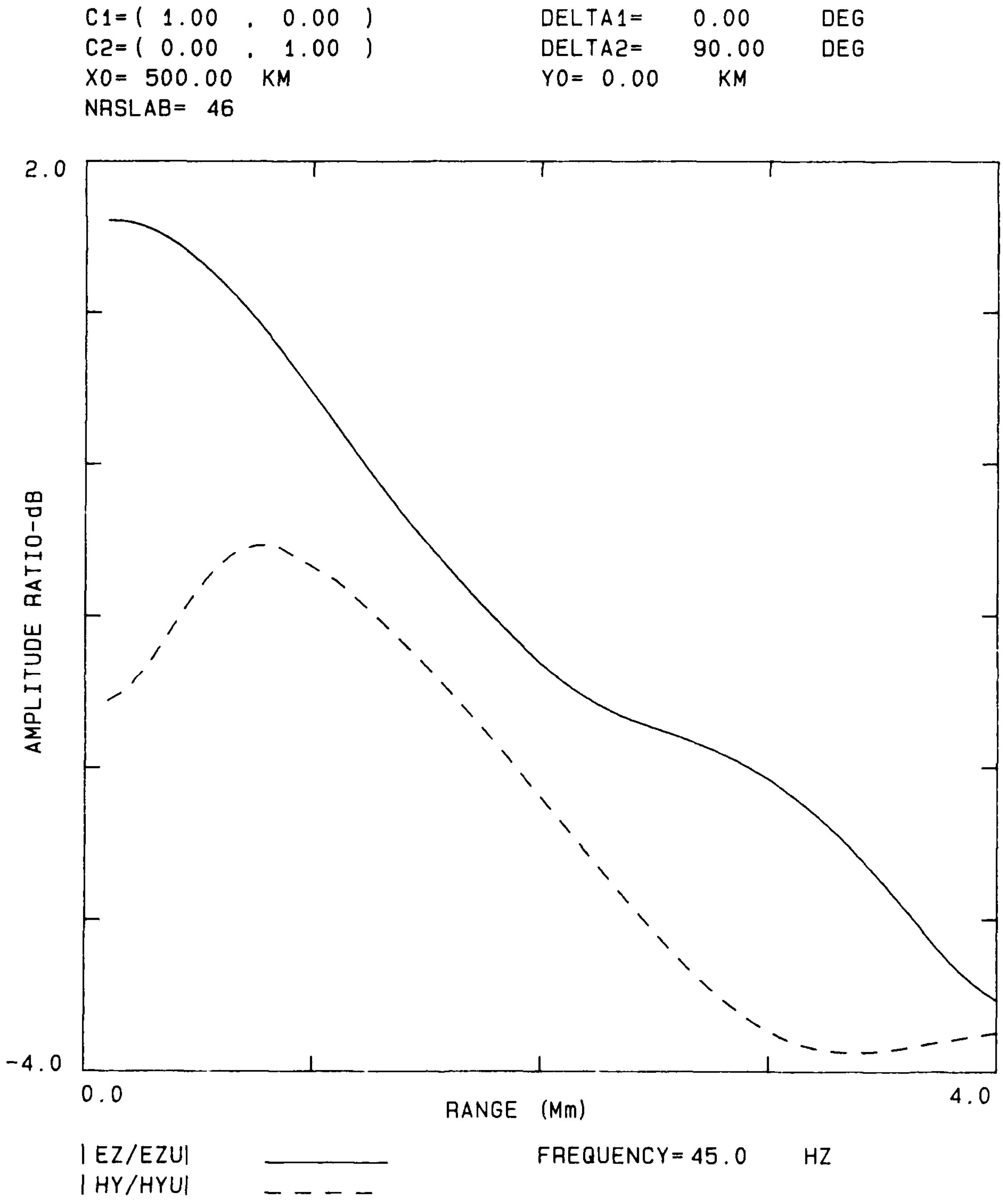


Figure 23. Amplitude ratios vs. range for a nuclear depression of radius 4.4 Mm. Disturbance center is on path and 500 km from transmitter.

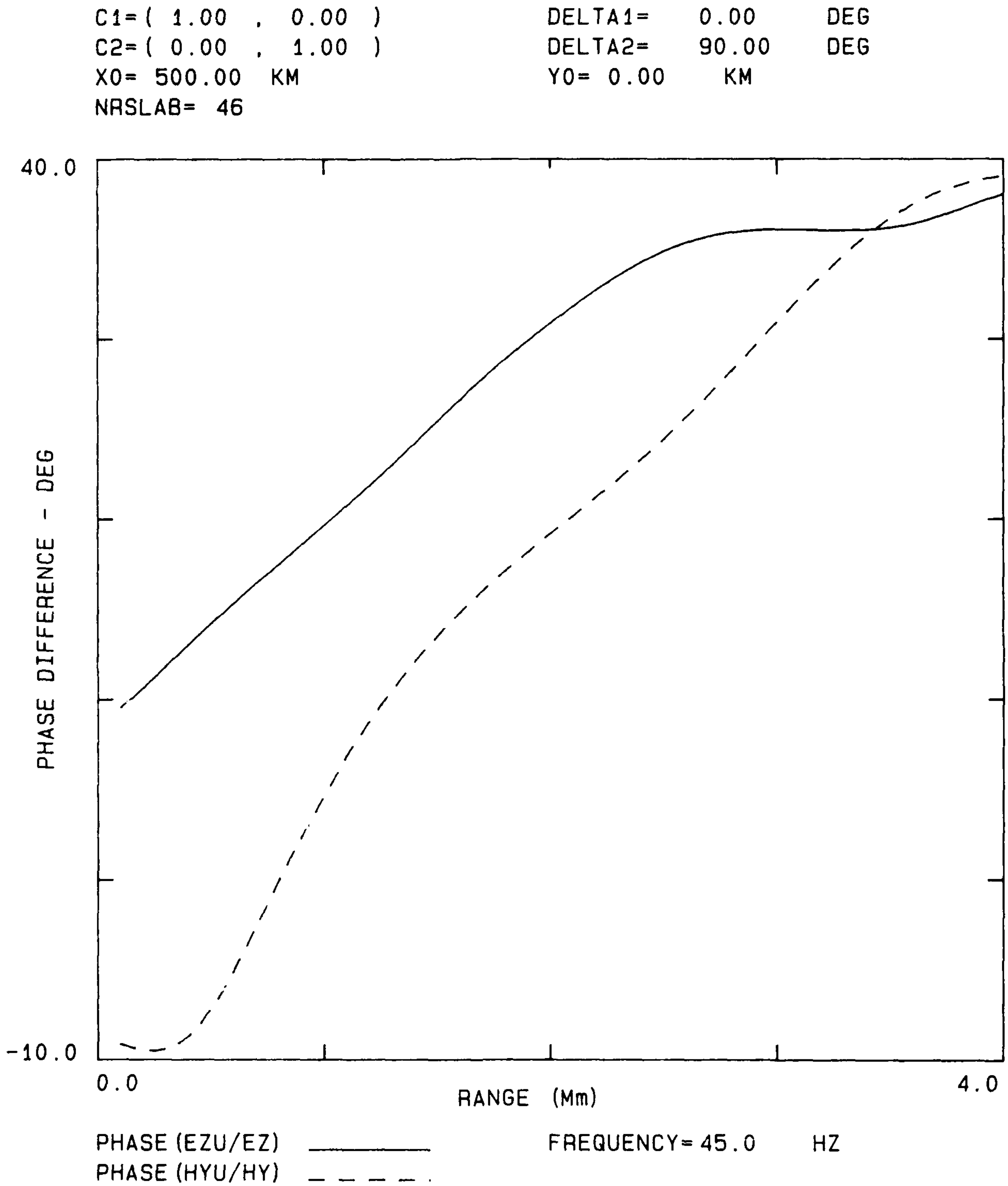


Figure 24. Phase differences vs. range for a nuclear depression of radius 4.4 Mm. Disturbance center is on path and 500 km from transmitter.

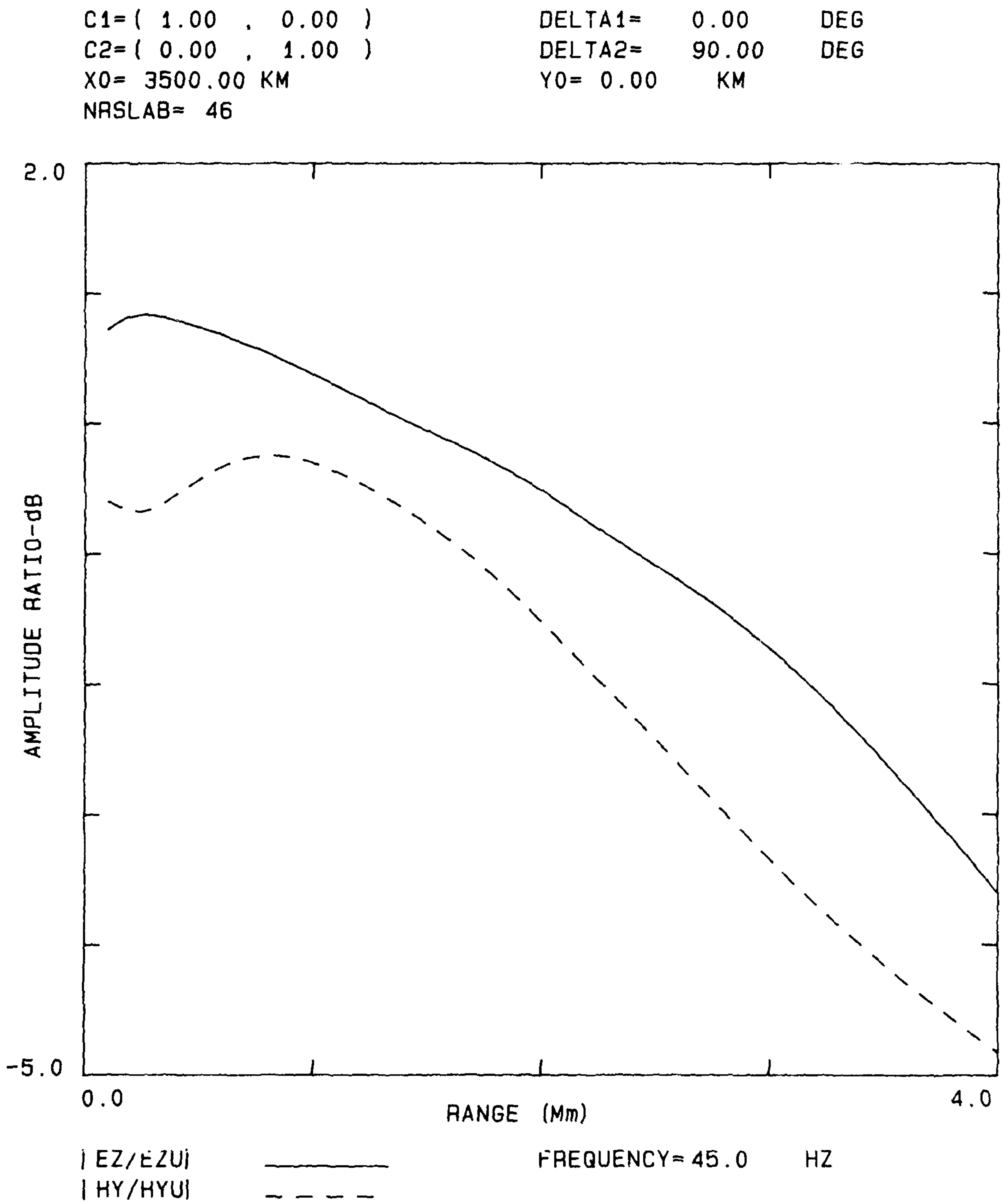


Figure 25. Amplitude ratios vs. range for a nuclear depression of radius 4.4 Mm. Disturbance center is on path and 3500 km from transmitter.

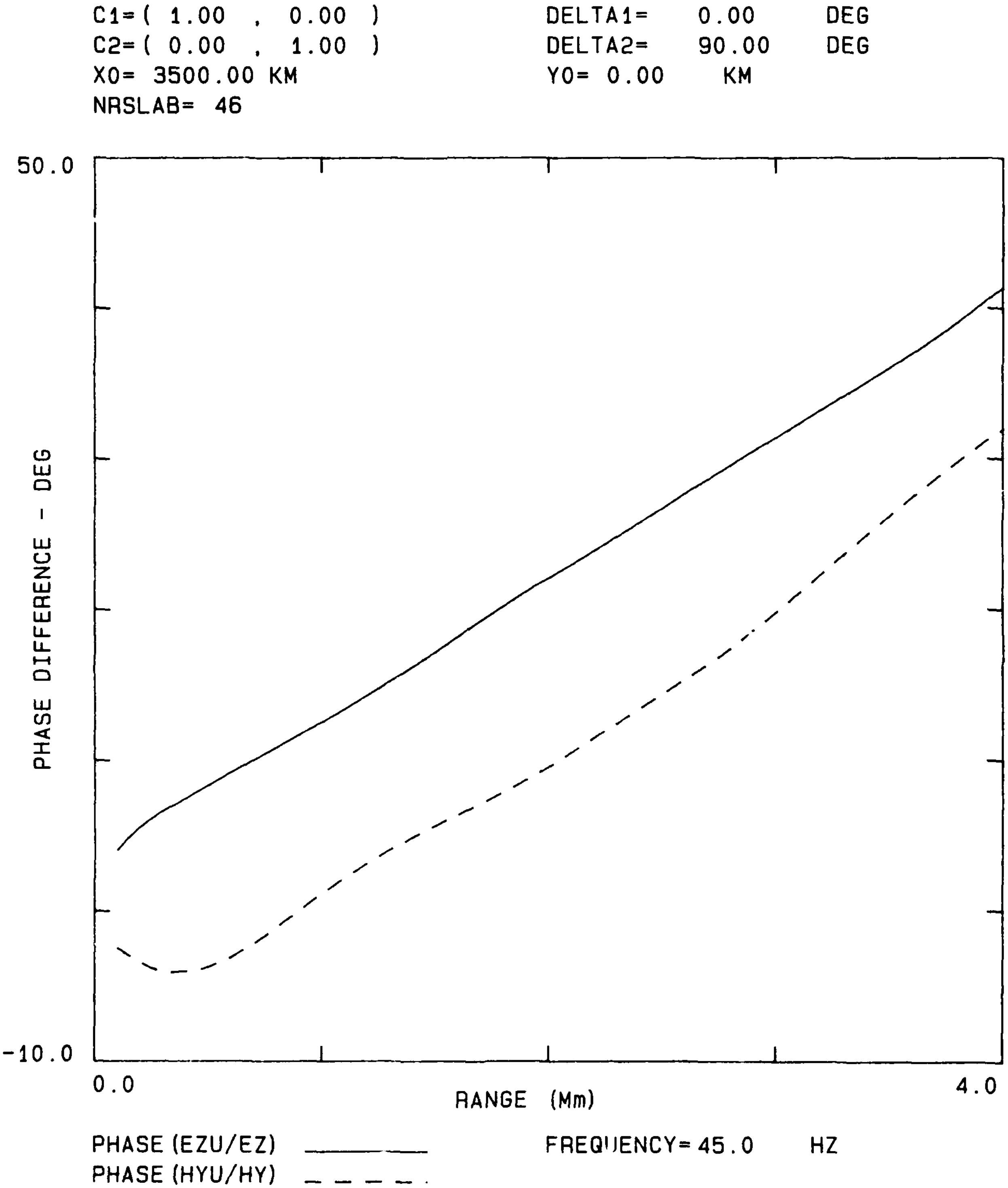


Figure 26. Phase differences vs. range for a nuclear depression of radius 4.4 Mm. Disturbance center is on path and 3500 km from transmitter.

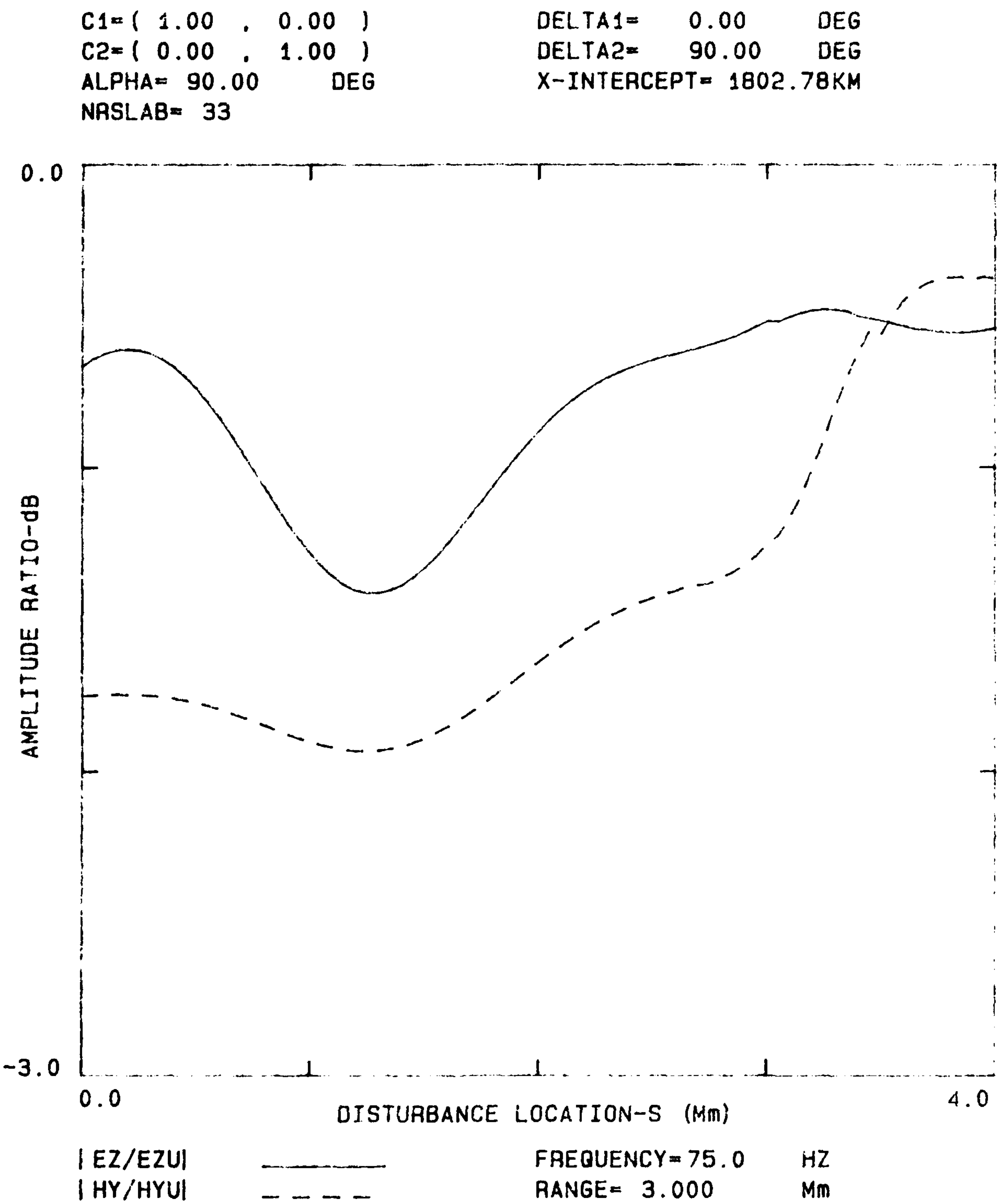


Figure 27. Amplitude ratios vs. disturbance location for a weak SPE. Polar cap boundary thickness = 1 Mm, range = 3 Mm.

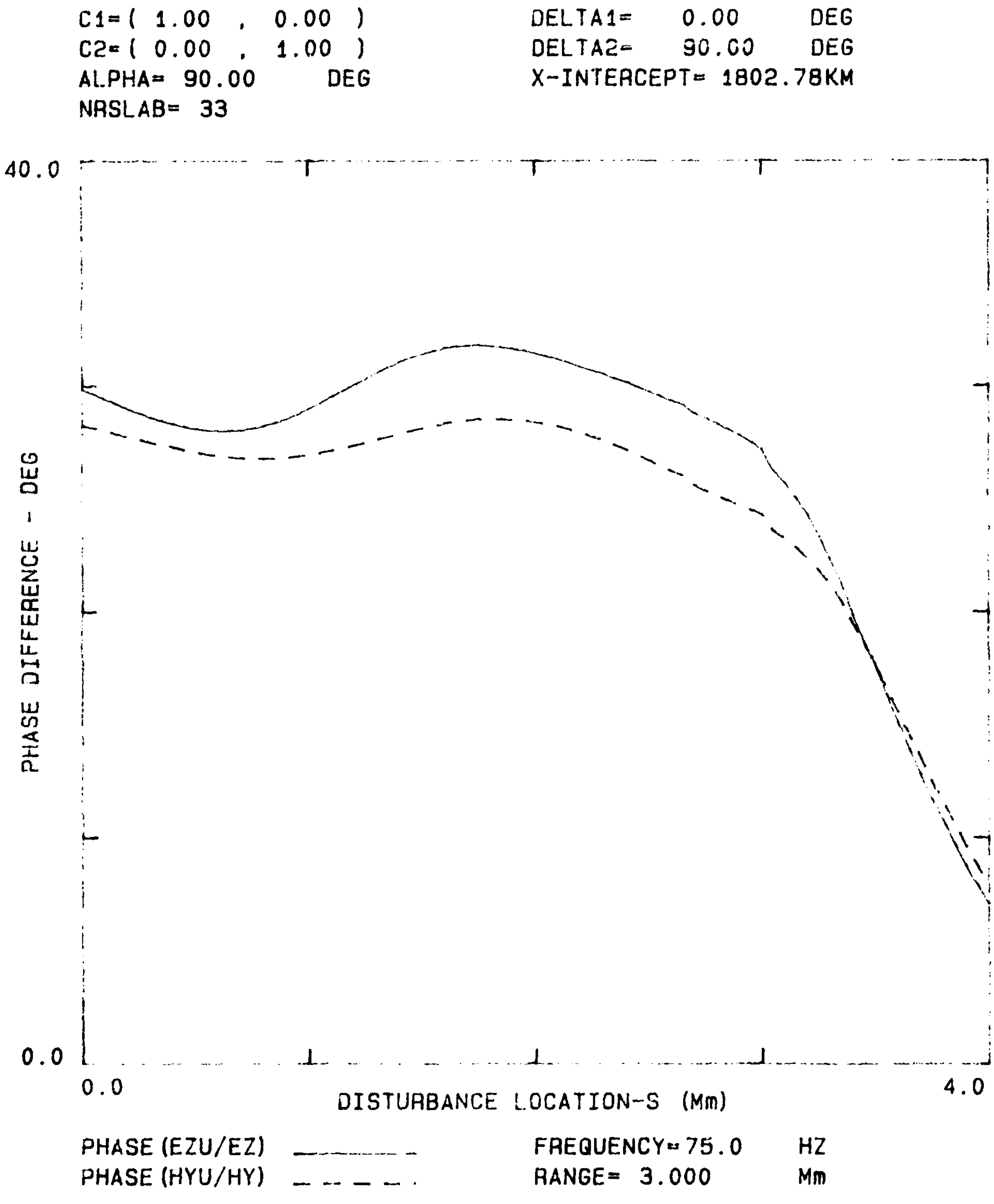


Figure 28. Phase differences vs. disturbance location for a weak SPE. Polar cap boundary thickness = 1 Mm, range = 3 Mm.

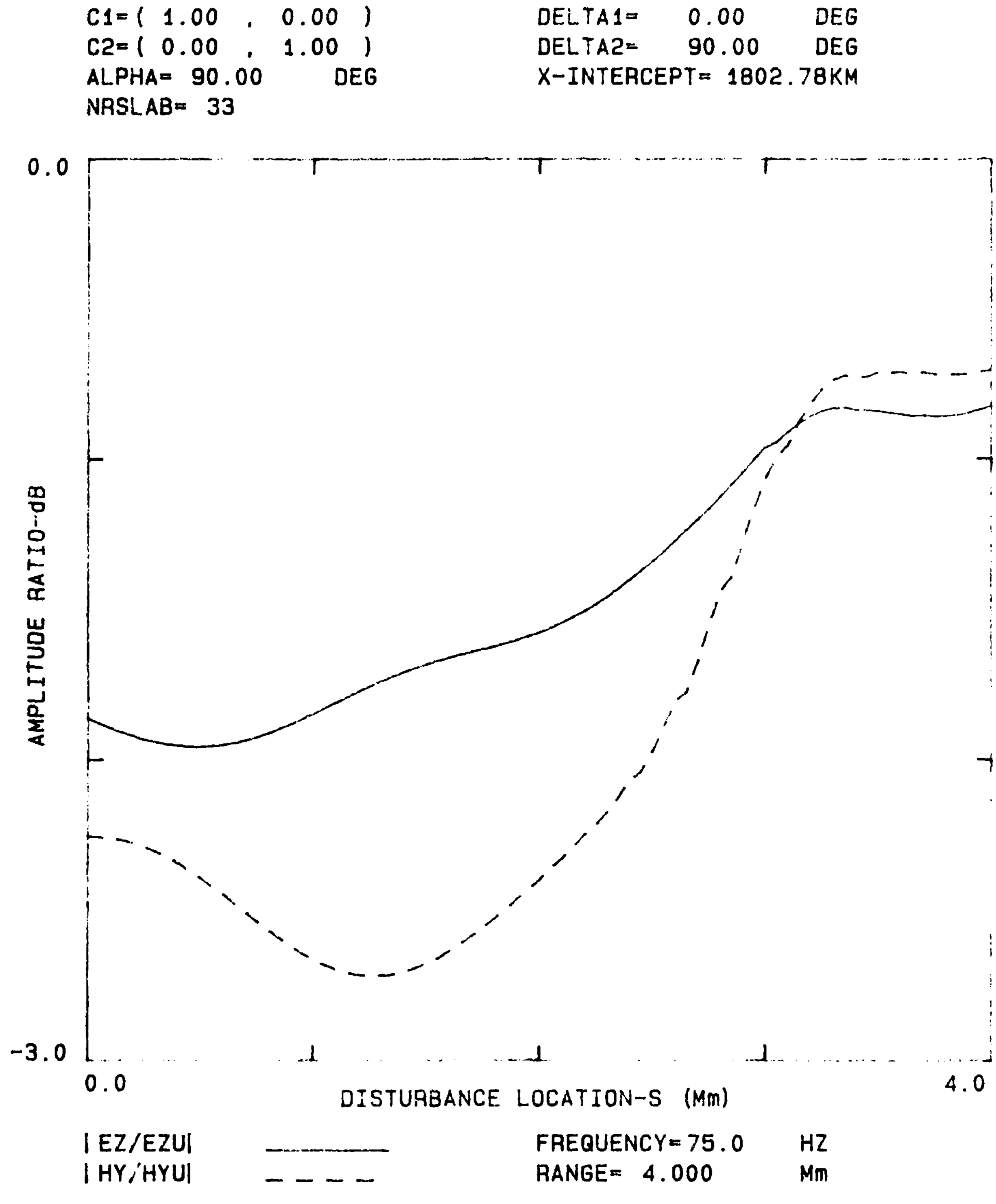


Figure 29. Amplitude ratios vs. disturbance location for a weak SPE. Polar cap boundary thickness = 1 Mm, range = 4 Mm.

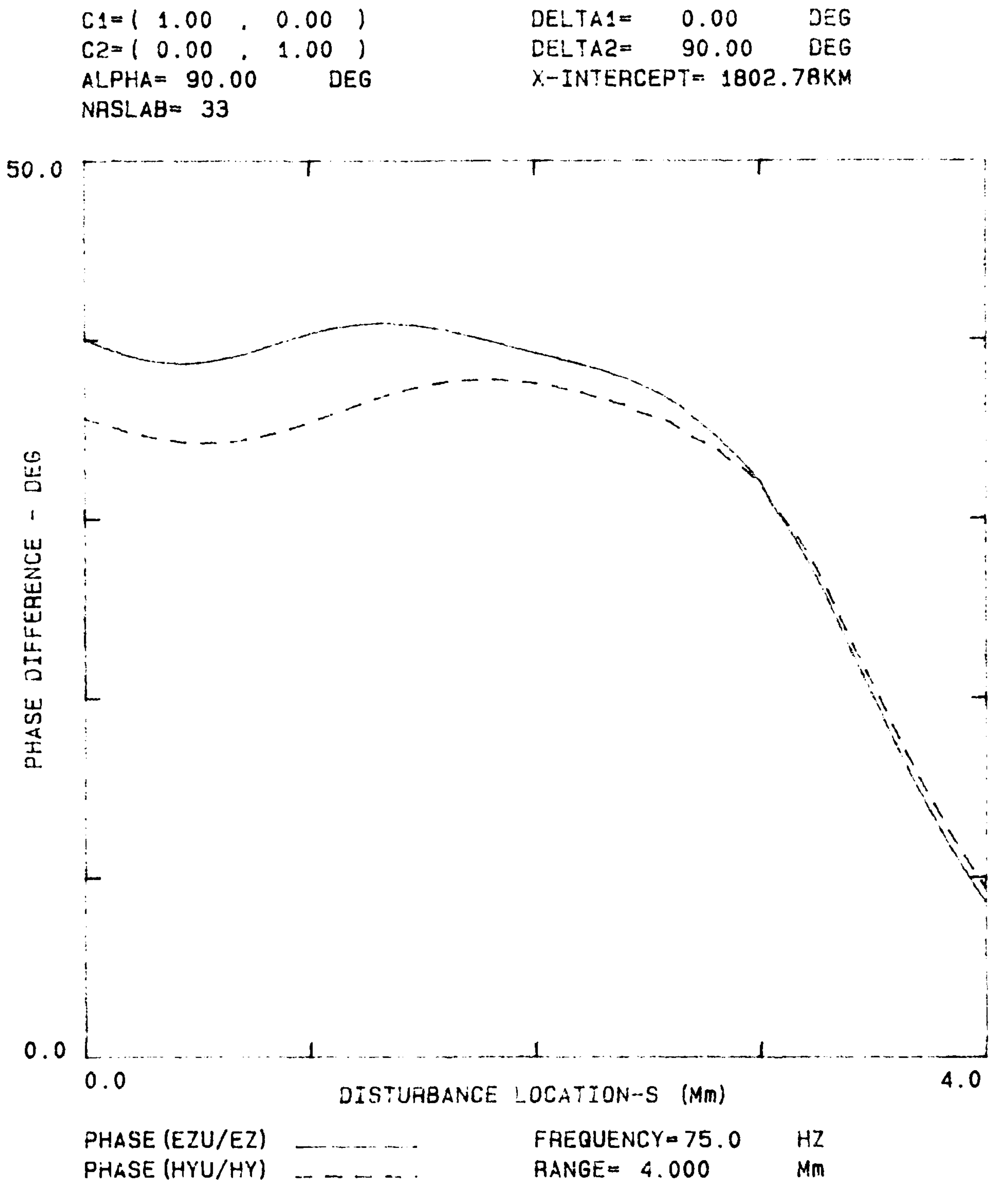


Figure 30. Phase differences vs. disturbance location for a weak SPE. Polar cap boundary thickness = 1 Mm, range = 4 Mm.

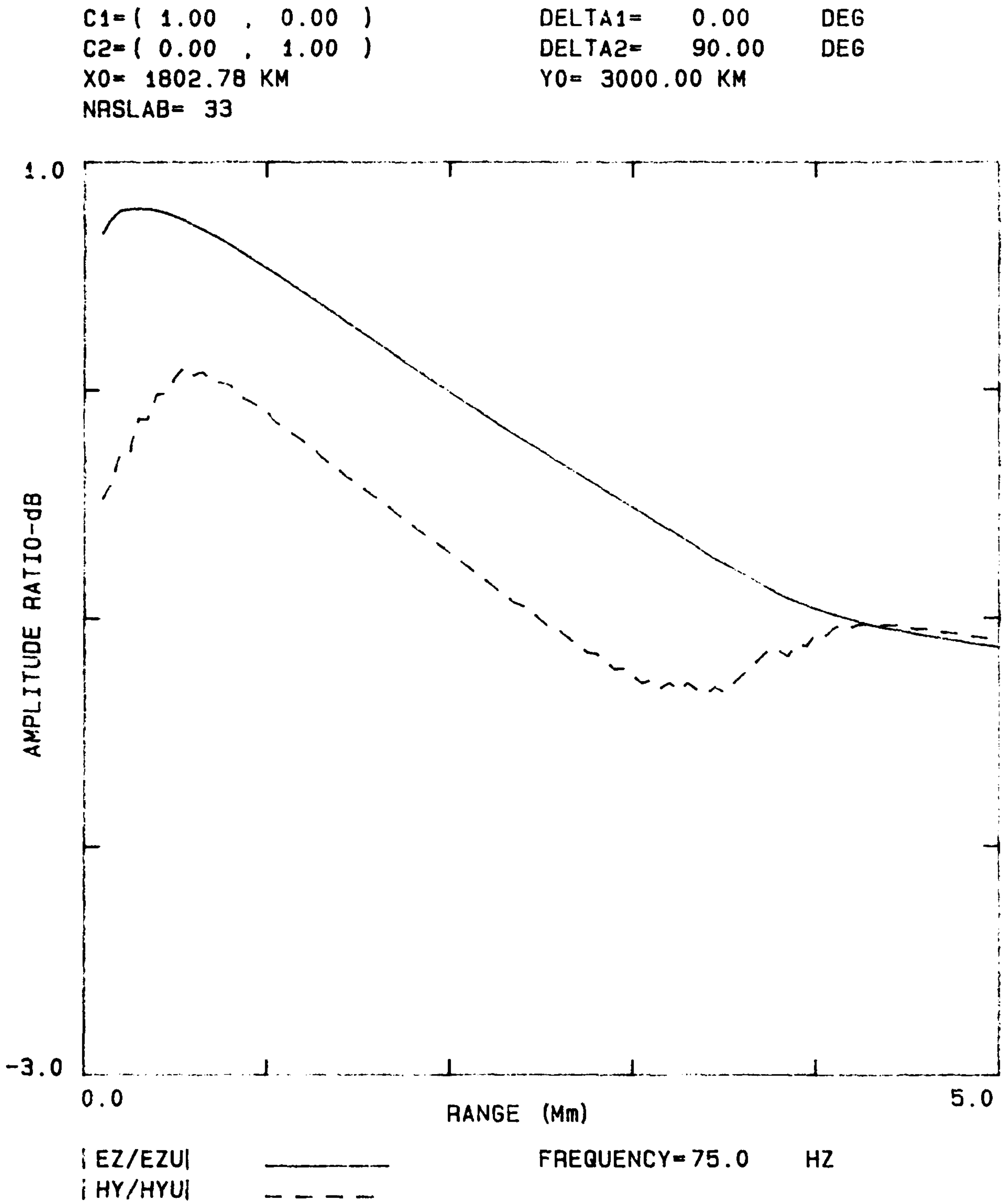


Figure 31. Amplitude ratios vs. range for grazing path on weak SPE polar cap boundary of thickness 1 Mm.

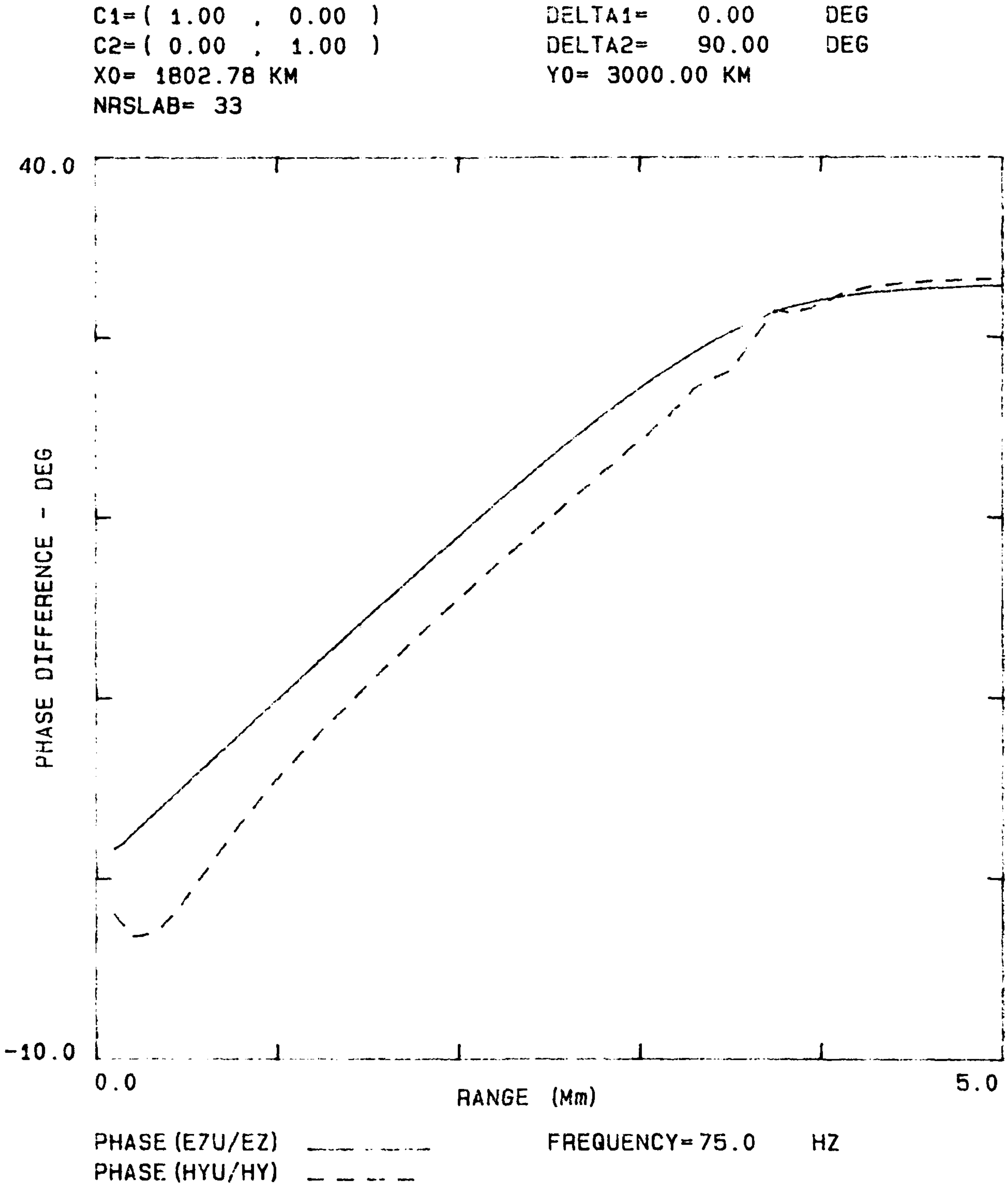


Figure 32. Phase differences vs. range for grazing path on weak SPE polar cap boundary of thickness 1 Mm.

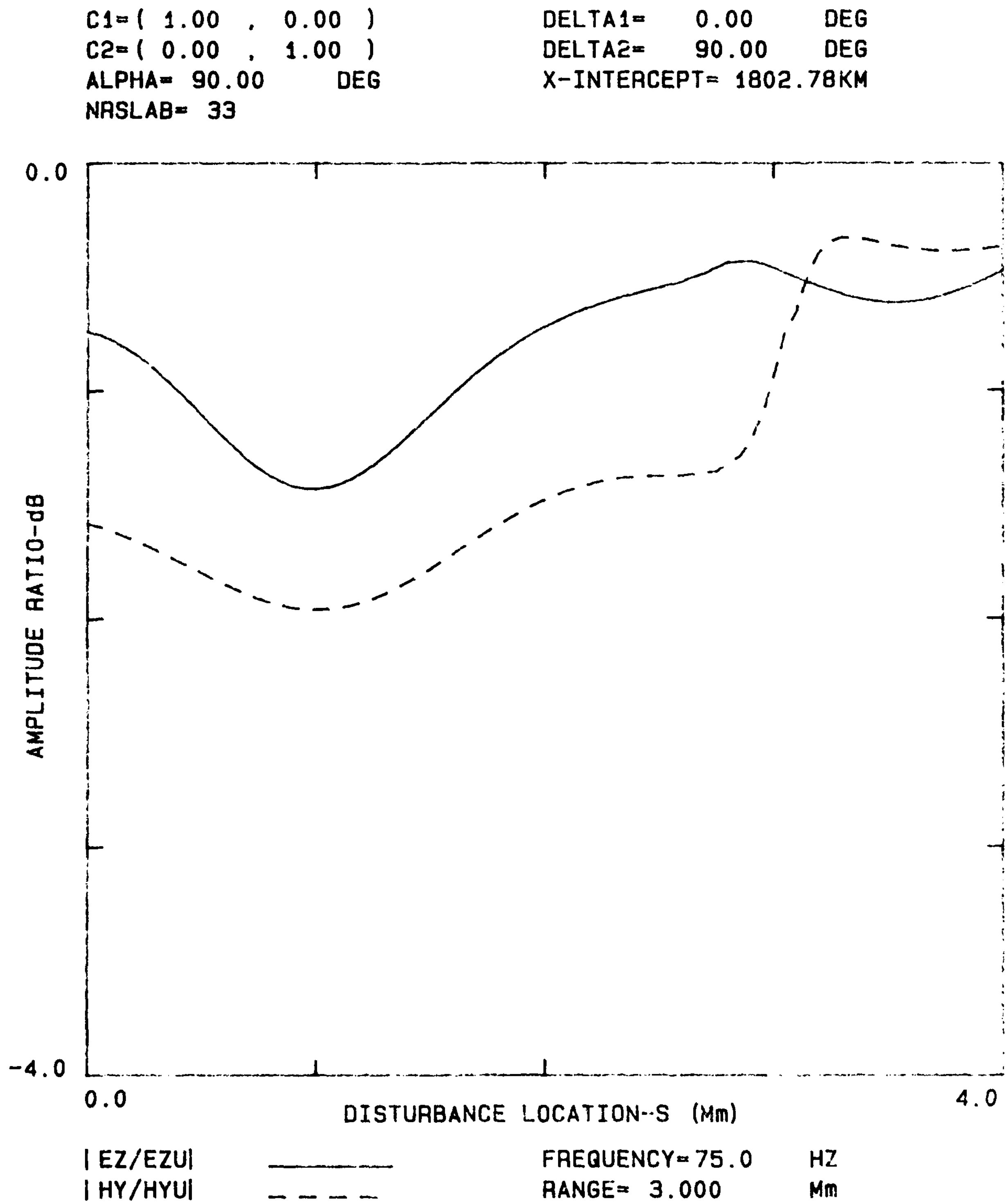


Figure 33. Amplitude ratios vs. disturbance location for a weak SPE. Polar cap boundary thickness = 0.5 Mm, range = 3 Mm.

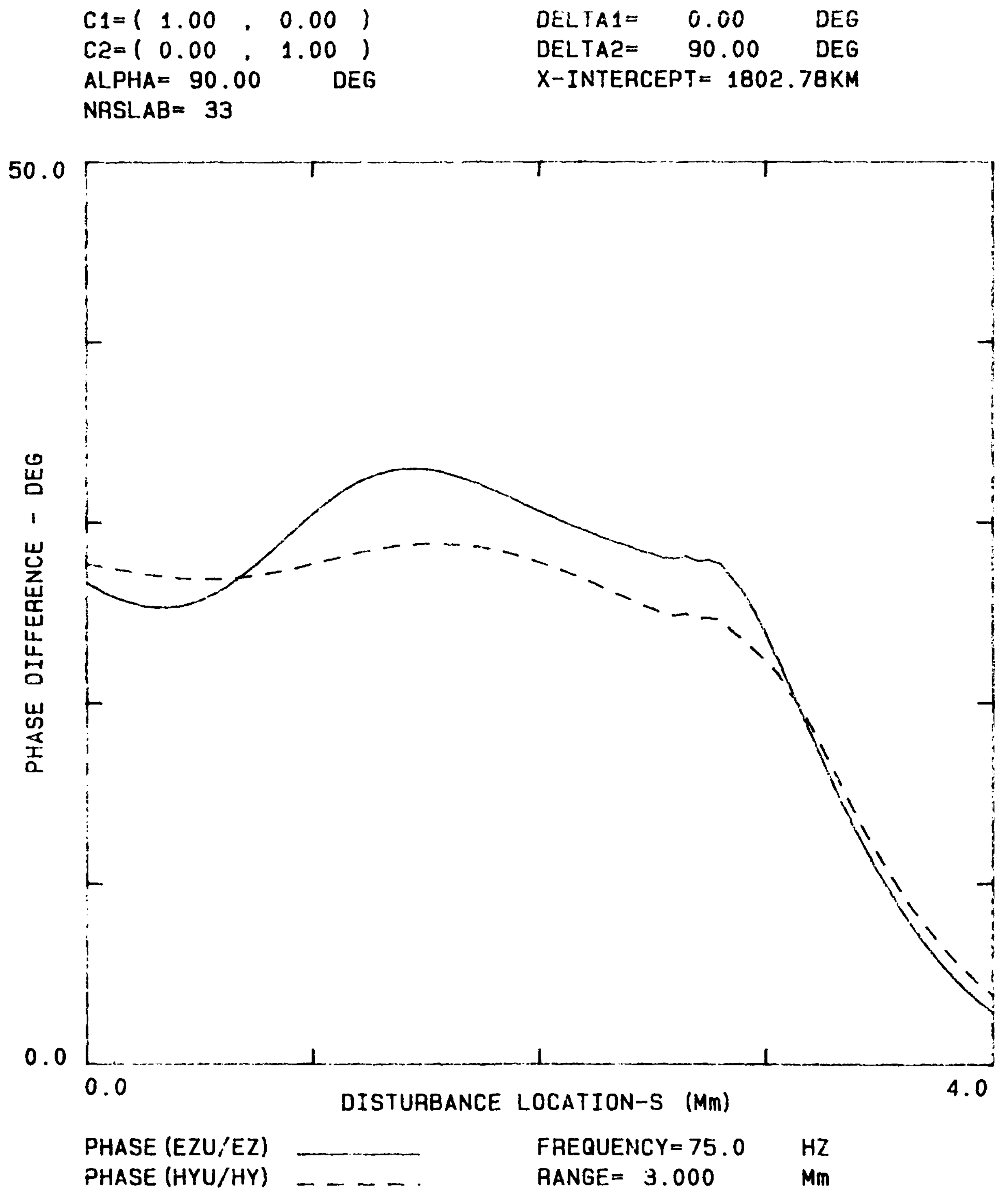


Figure 34. Phase differences vs. disturbance location for a weak SPE. Polar cap boundary thickness = 0.5 Mm, range = 3 Mm.

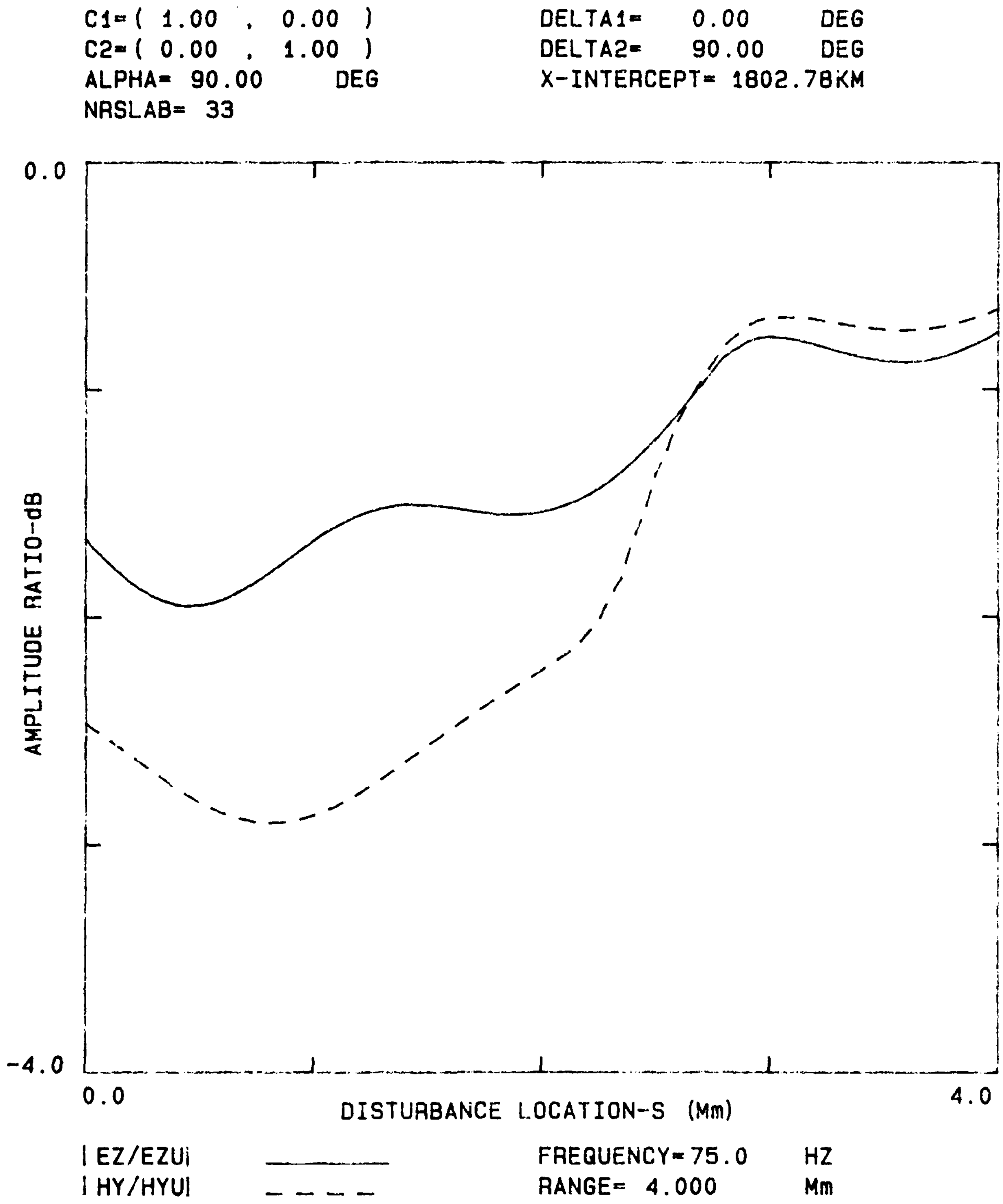


Figure 35. Amplitude ratios vs. disturbance location for a weak SPE. Polar cap boundary thickness = 0.5 Mm, range = 4 Mm.

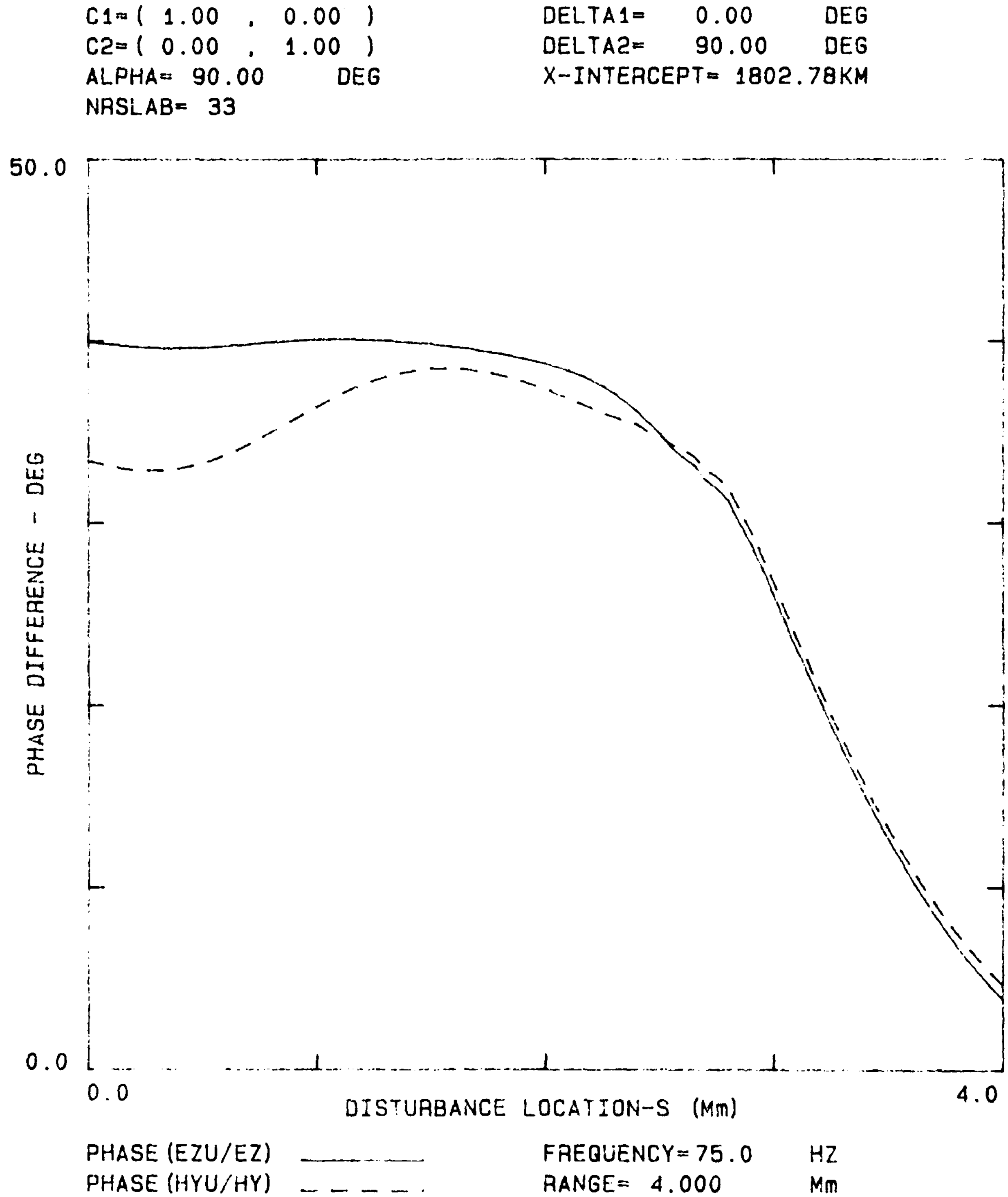


Figure 36. Phase differences vs. disturbance location for a weak SPE. Polar cap boundary thickness = 0.5 Mm, range = 4 Mm.

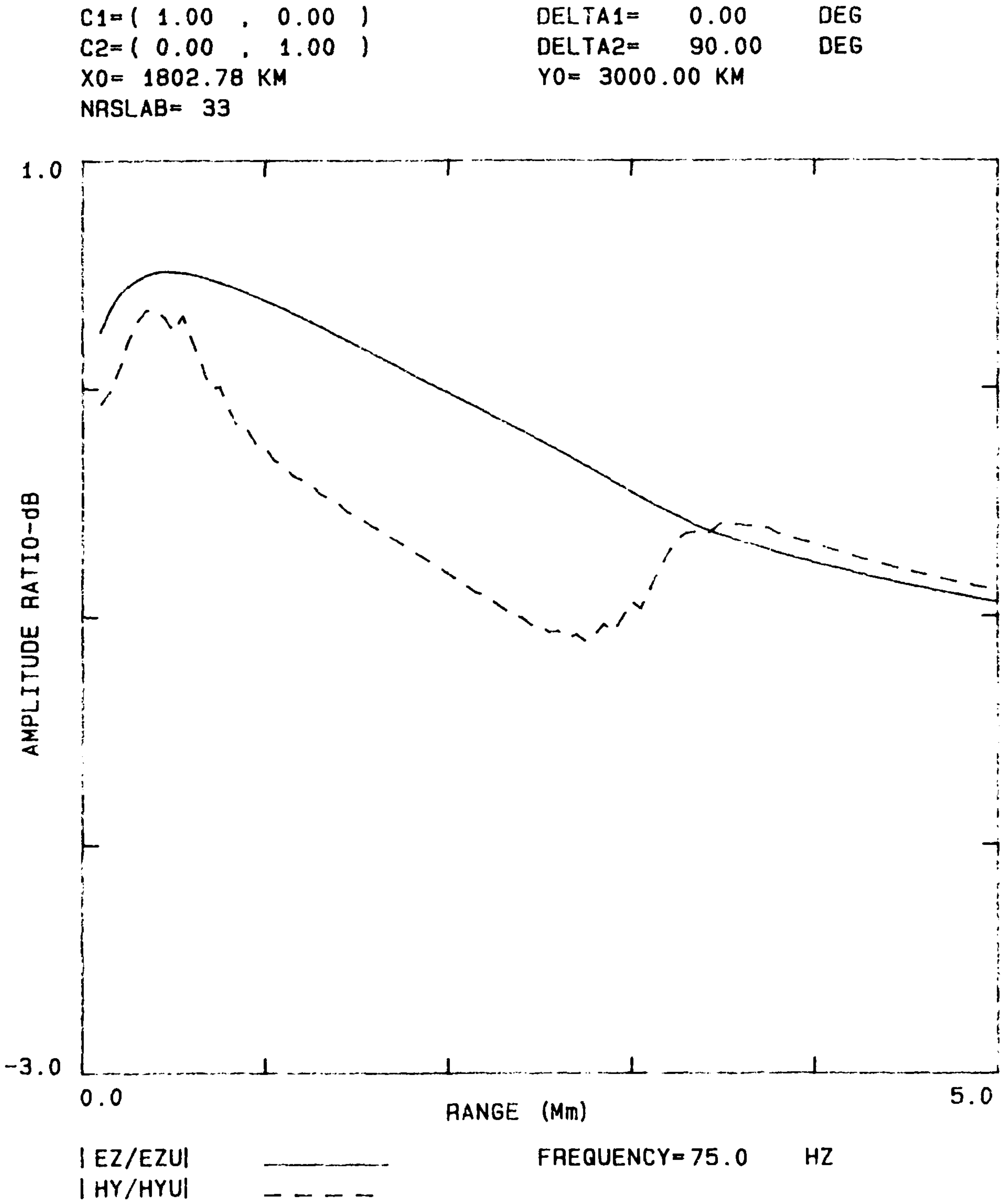


Figure 37. Amplitude ratios vs. range for grazing path on weak SPE polar cap boundary of thickness 0.5 Mm.

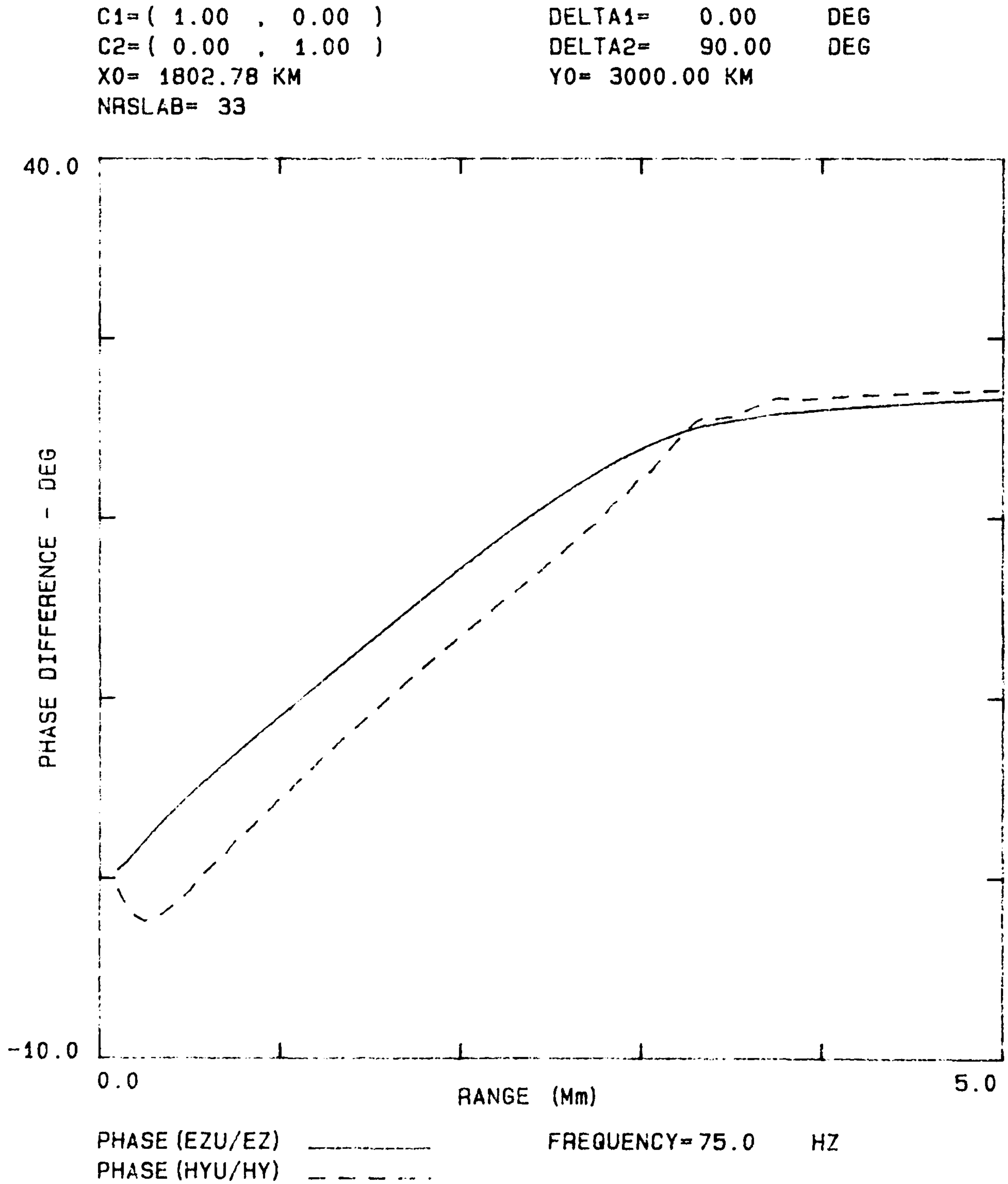


Figure 38. Phase differences vs. range for grazing path of weak SPE polar cap boundary of thickness 0.5 Mm.

C1= (1.00 , 0.00) DELTA1= 0.00 DEG
C2= (0.00 , 1.00) DELTA2= 90.00 DEG
X0= 3354.10 KM Y0= 1000.00 KM
NRSLAB= 33

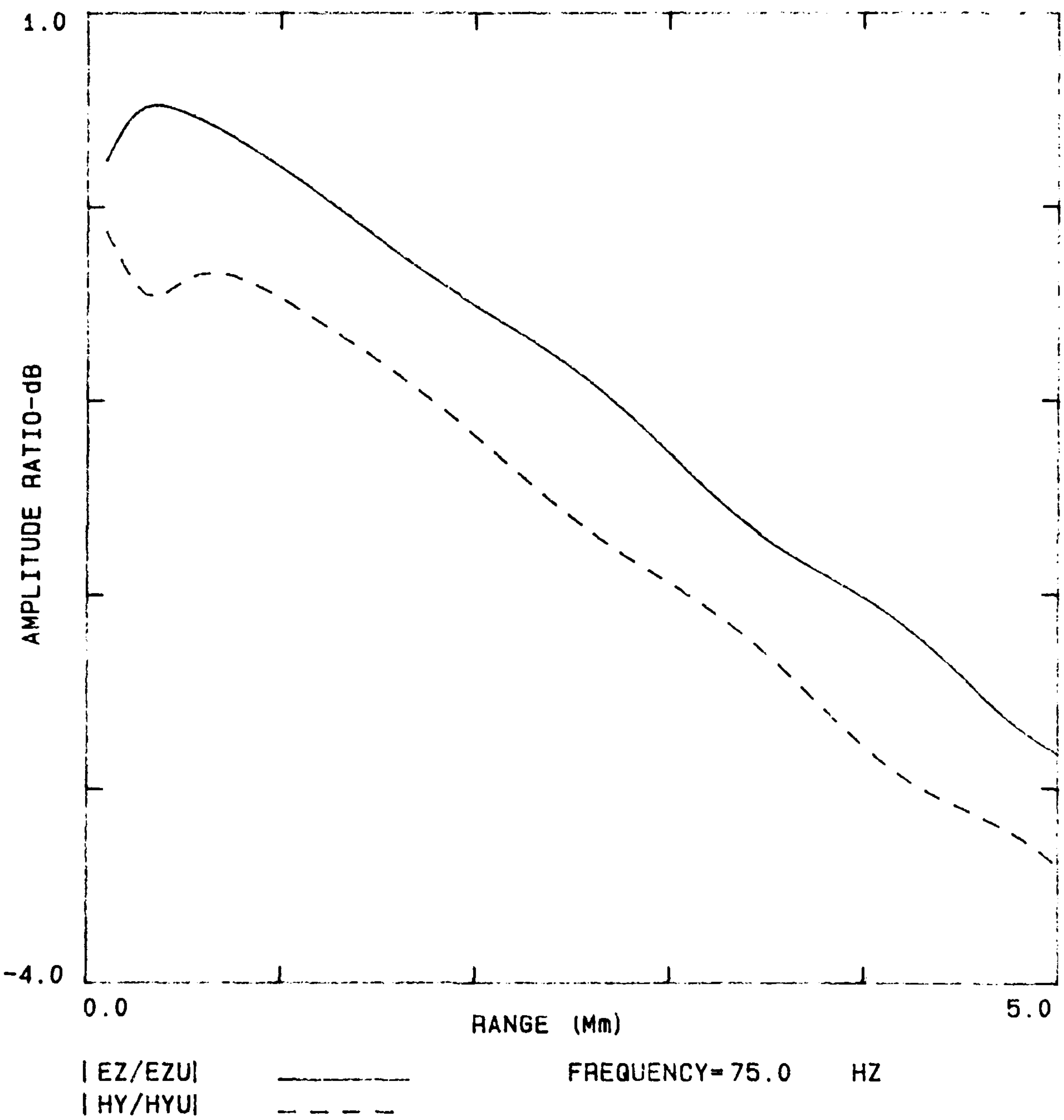


Figure 39. Amplitude ratios vs. range for penetrating polar cap path.

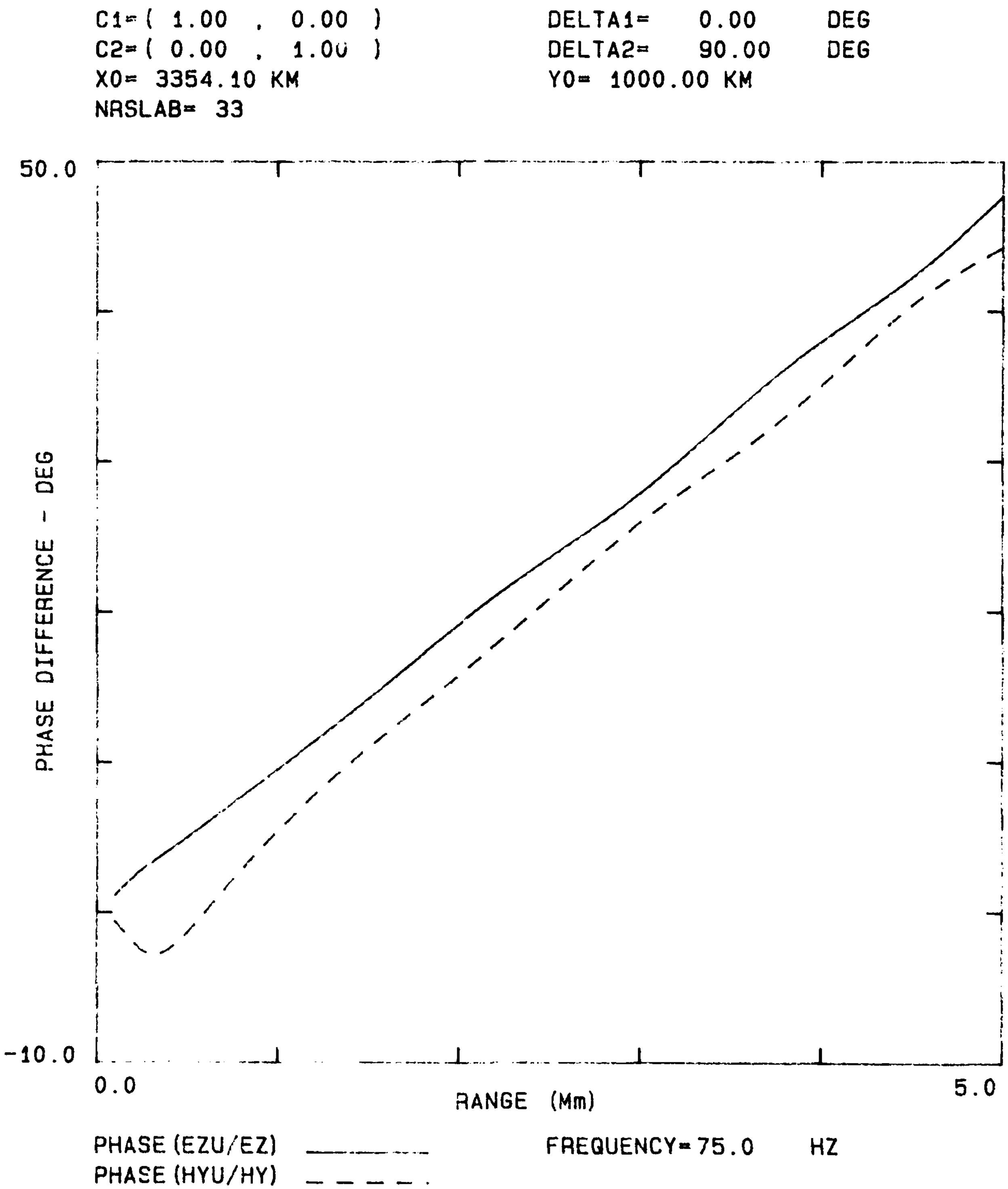


Figure 40. Phase differences vs. range for penetrating polar cap path.

APPENDIX A

ELF SCATTERING PROGRAM LISTING

```
c     main routine for program

      implicit real*8(a-h,o-z)
      open(unit=44,file='mlout')
      open(unit=40,file='mlplot',form='unformatted')
      call mlinit
      call mlcoef
      call mlflds
      stop
      end
```

```
      subroutine mlinit
      implicit real*8(a-h,o-z)
      complex*16 c1,c2,ampc,amps,thta,c,s,ssq,ks,r11b,r22b,t1,
     $lambda,f2,ksrho2,h2p,h2,bjp,bj,jrho2,jprho2,hrho2,hprho2,
     $ksrho1,jrho1,jprho1,hrho1,hprho1,num,den,gamma,d11,
     $ampprc,ampprs,gamma1
      character*16 fn
      parameter(mmax=51)
      parameter(mxslab=20)
      dimension sigma(mxslab),epsr(mxslab),rho(mxslab)
      dimension c(mxslab),s(mxslab),thta(mxslab),ssq(mxslab),
     $ks(mxslab),t1(mxslab),lambda(mxslab),f2(mxslab),ksrho2(mxslab),
     $h2p(0:mmax),h2(0:mmax),bjp(0:mmax),bj(0:mmax),jrho2(0:mmax,mxslab)
     $,jprho2(0:mmax,mxslab),hrho2(0:mmax,mxslab),hprho2(0:mmax,mxslab),
     $ksrho1(mxslab),jrho1(0:mmax,mxslab),jprho1(0:mmax,mxslab),
     $hrho1(0:mmax,mxslab),hprho1(0:mmax,mxslab),gamma(0:mmax,mxslab),
     $gamma1(0:mmax,mxslab)
      common/one/ampc,amps,ks,nrslab,r0,rho,s,ssq
      common/two/gamma,hprho1,hprho2,hrho1,hrho2,jprho1,jprho2,jrho1,
     $jrho2,gamma1
      common/three/alpha,ampprc,ampprs,c1,c2,cosalp,delta1,delta2,
     $iflag,rinc,rmax,rmin,sinalp,sinc,smax,smin,xyint,theta,x0,y0,skip
      data pi/3.1415926536/
      data vellit/2.997925e5/
      namelist/datum/rho,iflag,nrslab,rmin,rmax,rinc,smin,smax,sinc,
     $alpha,delta1,delta2,c1,c2,xyint,thta,frqkhz,skip
c       do n=1,nrslab
c         read in slab data
c      end do
c       print 101
c101    format('input filename:')
      read 102,fn
102   format(a16)
      open(unit=51,file=fn,status='old')
      read(51,datum)
      write(44,datum)
      write(40) iflag,nrslab,frqkhz,delta1,delta2,c1,c2,xyint,alpha
      freq=frqkhz*1000.
      waveno=2.*pi*freq/vellit
      cosalp=cos(alpha*1.7453292d-2)
      sinalp=sin(alpha*1.7453292d-2)
      if(alpha .eq. 0.)then
        x0=smin
        y0=xyint
      else
        x0=smin*cosalp+xyint
        y0=smin*sinalp
      end if
      r0=sqrt(x0**2+y0**2)
      if(r0 .lt. 1.e-4)then
        if(x0 .lt. 0.)then
          x0=-1.e-4
          r0=sqrt(x0**2+y0**2)
        else
          x0=1.e-4
          r0=sqrt(x0**2+y0**2)
        end if
```

```
        go to 2
      end if
2     if(x0 .eq. 0.)then
        theta=pi/2.
        if(y0 .lt. 0.)theta=-theta
      else
        theta=atan2(y0,x0)
      end if
c      write(44,*)'x0=',x0
c      write(44,*)'y0=',y0
      delta1=delta1*1.7453292e-2
      delta2=delta2*1.7453292e-2
      ctmd1=cos(theta-delta1)
      ctmd2=cos(theta-delta2)
      stmd1=sin(theta-delta1)
      stmd2=sin(theta-delta2)
      stheta=sin(theta)
      ctheta=cos(theta)
      beta1=-delta1
      beta2=-delta2
      sbeta1=sin(beta1)
      sbeta2=sin(beta2)
      cbeta1=cos(beta1)
      cbeta2=cos(beta2)
      ampc=2.*ctmd1*c1+2.*ctmd2*c2
      amps=2.*stmd1*c1+2.*stmd2*c2
      ampprc=c1*cbeta1+c2*cbeta2
      ampprs=-c1*sbeta1-c2*sbeta2
      do n=1,nrslab
        thta(n)=thta(n)*1.7453292e-2
        c(n)=zcos(thta(n))
        s(n)=zsin(thta(n))
        ssq(n)=s(n)*s(n)
        ks(n)=waveno*s(n)
c        call rbars(s(n),c(n),sigma(n),epsr(n),freq,r11b,r22b)
c        d11=(1.+r11b)*(1.+r11b)
c        lambda(n)=-d11*t1(n)*s(n)
c        f2(n)=c(n)*(1.-r11b)/(1.+r11b)

        if(n .eq. 1)then
          ksrho2(n)=ks(n)*rho(n)
          call hjfunc(ksrho2(n),bj,bjp,h2,h2p)
          do m=0,mmax-1
            jrho2(m,n)=bj(m)
            jprho2(m,n)=bjp(m)
            hrho2(m,n)=h2(m)
            hprho2(m,n)=h2p(m)
          end do
        else
          if(n .eq. nrslab)then
            ksrho1(n)=ks(n)*rho(n-1)
            call hjfunc(ksrho1(n),bj,bjp,h2,h2p)
            do m=0,mmax-1
              jrho1(m,n)=bj(m)
              jprho1(m,n)=bjp(m)
              hrho1(m,n)=h2(m)
              hprho1(m,n)=h2p(m)
            end do
```

```
          else
            ksrho2(n)=ks(n)*rho(n)
            ksrho1(n)=ks(n)*rho(n-1)
            call hjfunc(ksrho2(n),bj,bjp,h2,h2p)
            do m=0,mmax-1
              jrho2(m,n)=bj(m)
              jprho2(m,n)=bjp(m)
              hrho2(m,n)=h2(m)
              hprho2(m,n)=h2p(m)
            end do
            call hjfunc(ksrho1(n),bj,bjp,h2,h2p)
            do m=0,mmax-1
              jrho1(m,n)=bj(m)
              jprho1(m,n)=bjp(m)
              hrho1(m,n)=h2(m)
              hprho1(m,n)=h2p(m)
            end do
          end if
        end if
        if(n .ne. 1 .and. n .ne. nrslab)then
          if(n .eq. 2)then
            do m=0,mmax-1
              num=s(2)*jprho2(m,1)*jrho1(m,2)
              num=num-s(1)*jrho2(m,1)*jprho1(m,2)
              den=s(1)*jrho2(m,1)*hprho1(m,2)
              den=den-s(2)*jprho2(m,1)*hrho1(m,2)
              gamma(m,n)=num/den
            end do
          else
            do m=0,mmax-1
              num=s(n)*jrho1(m,n)*(jprho2(m,n-1)+gamma(m,n-1)
     $          *hprho2(m,n-1))
              num=num-s(n-1)*jprho1(m,n)*(jrho2(m,n-1)+gamma(m,n-1)
     $          *hrho2(m,n-1))
              den=s(n-1)*hprho1(m,n)*(jrho2(m,n-1)+gamma(m,n-1)
     $          *hrho2(m,n-1))
              den=den-s(n)*hrho1(m,n)*(jprho2(m,n-1)+gamma(m,n-1)*
     $          hprho2(m,n-1))
              gamma(m,n)=num/den
            end do
          end if
        end if
      end do
      do n=nrslab-1,2,-1
        if(n .eq. nrslab-1)then
          do m=0,mmax-1
            den=s(nrslab)*hrho1(m,nrslab)*jprho2(m,nrslab-1)
            den=den-s(nrslab-1)*hprho1(m,nrslab)*jrho2(m,nrslab-1)
            num=s(nrslab-1)*hprho1(m,nrslab)*hrho2(m,nrslab-1)
            num=num-s(nrslab)*hrho1(m,nrslab)*hprho2(m,nrslab-1)
            gamma1(m,nrslab-1)=num/den
          end do
        else
          do m=0,mmax-1
            num=s(n+1)*jprho2(m,n)*(gamma1(m,n+1)*jrho1(m,n+1)+
     $        hrho1(m,n+1))
            num=num-s(n)*jrho2(m,n)*(gamma1(m,n+1)*jprho1(m,n+1)+
     $        hprho1(m,n+1))
```

```
          den=s(n)*hrho2(m,n)*(gammal(m,n+1)*jprhol(m,n+1)+
     $          hprhol(m,n+1))
          den=den-s(n+1)*hprho2(m,n)*(gammal(m,n+1)*jrhol(m,n+1)+
     $          hrhol(m,n+1))
          gammal(m,n)=den/num
        end do
      end if
    end do
    return
    end
```

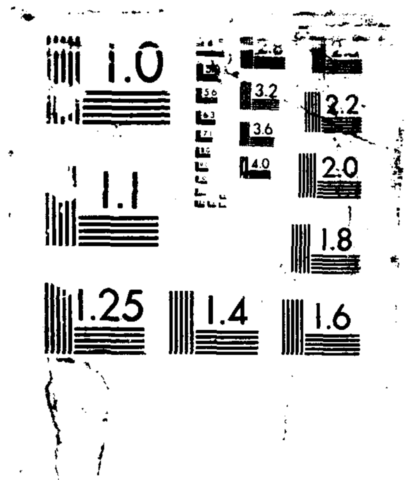
```
      subroutine mlcoef
      implicit complex*16(a-h,o-z)
      complex*16 ks,jr0,jpr0,lr0,jrhol,jrho2,jprhol,jprho2,num
      real*8 r0,rho
      parameter(mmax=51)
      parameter(mxslab=20)
      dimension rho(mxslab),ks(mxslab),jr0(0:mmax),jpr0(0:mmax),
     $hr0(0:mmax),hpr0(0:mmax),lr0(0:mmax),gr0(0:mmax),
     $jrhol(0:mmax,mxslab),jprhol(0:mmax,mxslab),jrho2(0:mmax,mxslab),
     $jprho2(0:mmax,mxslab),hrhol(0:mmax,mxslab),hprhol(0:mmax,mxslab),
     $hrho2(0:mmax,mxslab),hprho2(0:mmax,mxslab),ssq(mxslab),s(mxslab),
     $a2(2,2),b2(2),x2(2),a4(4,4),b4(4),x4(4),xjc(0:mmax,mxslab),
     $xjs(0:mmax,mxslab),xhc(0:mmax,mxslab),xhs(0:mmax,mxslab),
     $gamma(0:mmax,mxslab),a2sav(2,2),a4sav(4,4),gammal(0:mmax,mxslab)
      common/one/ampc,amps,ks,nrslab,r0,rho,s,ssq
      common/two/gamma,hprhol,hprho2,hrhol,hrho2,jprhol,jprho2,jrhol,
     $jrho2,gammal
      common/four/xhc,xhs,xjc,xjs
      common/five/gr0,hpr0,hr0,jpr0,jr0,lr0
      nt=nrslab
      do n=1,nrslab-1
        if(r0 .le. rho(n))then
          nt=n
          go to 1
        end if
      end do
1     continue
      arg=ks(nt)*r0
      call hjfunc(arg,jr0,jpr0,hr0,hpr0)
      do m=0,mmax-1
        lr0(m)=jr0(m)/arg
        gr0(m)=hr0(m)/arg
      end do
      if(nt .eq. 1 .and. nrslab .eq. 2)then
        do m=0,mmax-1
          a2(1,1)=ssq(1)*jrho2(m,1)
          a2(1,2)=-ssq(2)*hrhol(m,2)
          a2(2,1)=s(1)*jprho2(m,1)
          a2(2,2)=-s(2)*hprhol(m,2)
          a2sav(1,1)=a2(1,1)
          a2sav(1,2)=a2(1,2)
          a2sav(2,1)=a2(2,1)
          a2sav(2,2)=a2(2,2)
          b2(1)=ampc*ssq(1)*jpr0(m)*hrho2(m,1)
          b2(2)=ampc*s(1)*jpr0(m)*hprho2(m,1)
          if(m .eq. 0)then
            b2(1)=.5*b2(1)
            b2(2)=.5*b2(2)
          end if
          call clineq(a2,b2,x2,2,2,0,err)
          xjc(m,1)=x2(1)
          xhc(m,2)=x2(2)
          b2(1)=-amps*ssq(1)*m*lr0(m)*hrho2(m,1)
          b2(2)=-amps*s(1)*m*lr0(m)*hprho2(m,1)
          call clineq(a2sav,b2,x2,2,2,0,err)
          xjs(m,1)=x2(1)
          xhs(m,2)=x2(2)
```

```
      end do
      go to 2
    end if
    if(nt .eq. 2 .and. nrslab .eq. 2)then
      do m=0,mmax-1
        a2(1,1)=-ssq(1)*jrho2(m,1)
        a2(1,2)=ssq(2)*hrho1(m,2)
        a2(2,1)=-s(1)*jprho2(m,1)
        a2(2,2)=s(2)*hprho1(m,2)
        a2sav(1,1)=a2(1,1)
        a2sav(1,2)=a2(1,2)
        a2sav(2,1)=a2(2,1)
        a2sav(2,2)=a2(2,2)
        b2(1)=ssq(2)*ampc*hpr0(m)*jrho1(m,2)
        b2(2)=s(2)*ampc*hpr0(m)*jprho1(m,2)
        if(m .eq. 0)then
          b2(1)=.5*b2(1)
          b2(2)=.5*b2(2)
        end if
        call clineq(a2,b2,x2,2,2,0,err)
        xjc(m,1)=x2(1)
        xhc(m,2)=x2(2)
        b2(1)=-amps*ssq(2)*m*gr0(m)*jrho1(m,2)
        b2(2)=-amps*s(2)*m*gr0(m)*jprho1(m,2)
        call clineq(a2sav,b2,x2,2,2,0,err)
        xjs(m,1)=x2(1)
        xhs(m,2)=x2(2)
      end do
      go to 2
    end if
    if(nt .eq. 2 .and. nrslab .eq. 3)then
      do m=0,mmax-1
        a4(1,1)=-ssq(1)*jrho2(m,1)
        a4(1,2)=ssq(2)*jrho1(m,2)
        a4(1,3)=ssq(2)*hrho1(m,2)
        a4(1,4)=0.
        a4(2,1)=-s(1)*jprho2(m,1)
        a4(2,2)=s(2)*jprho1(m,2)
        a4(2,3)=s(2)*hprho1(m,2)
        a4(2,4)=0.
        a4(3,1)=0.
        a4(3,2)=ssq(2)*jrho2(m,2)
        a4(3,3)=ssq(2)*hrho2(m,2)
        a4(3,4)=-ssq(3)*hrho1(m,3)
        a4(4,1)=0.
        a4(4,2)=s(2)*jprho2(m,2)
        a4(4,3)=s(2)*hprho2(m,2)
        a4(4,4)=-s(3)*hprho1(m,3)
        a4sav(1,1)=a4(1,1)
        a4sav(1,2)=a4(1,2)
        a4sav(1,3)=a4(1,3)
        a4sav(1,4)=a4(1,4)
        a4sav(2,1)=a4(2,1)
        a4sav(2,2)=a4(2,2)
        a4sav(2,3)=a4(2,3)
        a4sav(2,4)=a4(2,4)
        a4sav(3,1)=a4(3,1)
        a4sav(3,2)=a4(3,2)
```

```
      a4sav(3,3)=a4(3,3)
      a4sav(3,4)=a4(3,4)
      a4sav(4,1)=a4(4,1)
      a4sav(4,2)=a4(4,2)
      a4sav(4,3)=a4(4,3)
      a4sav(4,4)=a4(4,4)
      b4(1)=ampc*ssq(2)*hpr0(m)*jrho1(m,2)
      b4(2)=ampc*s(2)*hpr0(m)*jprho1(m,2)
      b4(3)=ampc*ssq(2)*jpr0(m)*hrho2(m,2)
      b4(4)=ampc*s(2)*jpr0(m)*hprho2(m,2)
      if(m .eq. 0)then
        b4(1)=.5*b4(1)
        b4(2)=.5*b4(2)
        b4(3)=.5*b4(3)
        b4(4)=.5*b4(4)
      end if
      call clineq(a4,b4,x4,4,4,0,err)
      xjc(m,1)=x4(1)
      xjc(m,2)=x4(2)
      xhc(m,2)=x4(3)
      xhc(m,3)=x4(4)
      b4(1)=-amps*ssq(2)*m*gr0(m)*jrho1(m,2)
      b4(2)=-amps*s(2)*m*gr0(m)*jprho1(m,2)
      b4(3)=-amps*ssq(2)*m*lr0(m)*hrho2(m,2)
      b4(4)=-amps*s(2)*m*lr0(m)*hprho2(m,2)
      call clineq(a4sav,b4,x4,4,4,0,err)
      xjs(m,1)=x4(1)
      xjs(m,2)=x4(2)
      xhs(m,2)=x4(3)
      xhs(m,3)=x4(4)
    end do
    go to 2
  end if
  if(nt .eq. 1 .and. nrslab .ge. 3)then
    do m=0,mmax-1
      a2(1,1)=ssq(1)*jrho2(m,1)
      a2(1,2)=-ssq(2)*(gamma1(m,2)*jrho1(m,2)+hrho1(m,2))
      a2(2,1)=s(1)*jprho2(m,1)
      a2(2,2)=-s(2)*(gamma1(m,2)*jprho1(m,2)+hprho1(m,2))
      a2sav(1,1)=a2(1,1)
      a2sav(1,2)=a2(1,2)
      a2sav(2,1)=a2(2,1)
      a2sav(2,2)=a2(2,2)
      b2(1)=ampc*ssq(1)*jpr0(m)*hrho2(m,1)
      b2(2)=ampc*s(1)*jpr0(m)*hprho2(m,1)
      if(m .eq. 0)then
        b2(1)=.5*b2(1)
        b2(2)=.5*b2(2)
      end if
      call clineq(a2,b2,x2,2,2,0,err)
      xjc(m,1)=x2(1)
      xhc(m,2)=x2(2)
      b2(1)=-amps*ssq(1)*m*lr0(m)*hrho2(m,1)
      b2(2)=-amps*s(1)*m*lr0(m)*hprho2(m,1)
      call clineq(a2sav,b2,x2,2,2,0,err)
      xjs(m,1)=x2(1)
      xhs(m,2)=x2(2)
      xjc(m,2)=gamma1(m,2)*xhc(m,2)
```

ELF SCATTERING IN THE EARTH-IONOSPHERE WAVEGUIDE(U) 2/2
NAVAL OCEAN SYSTEMS CENTER SAN DIEGO CA
R A PAPPERT ET AL. APR 88 NOSC/TD-1275

UNCLASSIFIED F/G 20/14 NL


1.0
2.8
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

```
        xjs(m,2)=gammal(m,2)*xhs(m,2)
        do n=3,nrslab-1
          num=ssq(n-1)*(gammal(m,n-1)*jrho2(m,n-1)+hrho2(m,n-1))
          den=ssq(n)*(gammal(m,n)*jrhol(m,n)+hrhol(m,n))
          xhc(m,n)=num*xhc(m,n-1)/den
          xjc(m,n)=gammal(m,n)*xhc(m,n)
          xhs(m,n)=num*xhs(m,n-1)/den
          xjs(m,n)=gammal(m,n)*xhs(m,n)
        end do
        num=ssq(nrslab-1)*(gammal(m,nrslab-1)*jrho2(m,nrslab-1)+
$           hrho2(m,nrslab-1))
        den=ssq(nrslab)*hrhol(m,nrslab)
        xhc(m,nrslab)=num*xhc(m,nrslab-1)/den
        xhs(m,nrslab)=num*xhs(m,nrslab-1)/den
      end do
      go to 2
    end if
    if(nt .eq. nrslab .and. nrslab .ge. 3)then
      do m=0,mmax-1
        a2(1,1)=-ssq(nrslab-1)*(jrho2(m,nrslab-1)+gamma(m,nrslab-1)*
$               hrho2(m,nrslab-1))
        a2(1,2)=ssq(nrslab)*hrhol(m,nrslab)
        a2(2,1)=-s(nrslab-1)*(jprho2(m,nrslab-1)+gamma(m,nrslab-1)*
$               hprho2(m,nrslab-1))
        a2(2,2)=s(nrslab)*hprhol(m,nrslab)
        a2sav(1,1)=a2(1,1)
        a2sav(1,2)=a2(1,2)
        a2sav(2,1)=a2(2,1)
        a2sav(2,2)=a2(2,2)
        b2(1)=ssq(nrslab)*ampc*hpr0(m)*jrhol(m,nrslab)
        b2(2)=s(nrslab)*ampc*hpr0(m)*jprhol(m,nrslab)
        if(m .eq. 0)then
          b2(1)=.5*b2(1)
          b2(2)=.5*b2(2)
        end if
        call clineq(a2,b2,x2,2,2,0,err)
        xjc(m,nrslab-1)=x2(1)
        xhc(m,nrslab)=x2(2)
        b2(1)=-ssq(nrslab)*amps*m*gr0(m)*jrhol(m,nrslab)
        b2(2)=-s(nrslab)*amps*m*gr0(m)*jprhol(m,nrslab)
        call clineq(a2sav,b2,x2,2,2,0,err)
        xjs(m,nrslab-1)=x2(1)
        xhs(m,nrslab)=x2(2)
        xhc(m,nrslab-1)=gamma(m,nrslab-1)*xjc(m,nrslab-1)
        xhs(m,nrslab-1)=gamma(m,nrslab-1)*xjs(m,nrslab-1)
        do n=nrslab-2,2,-1
          num=ssq(n+1)*(jrhol(m,n+1)+gamma(m,n+1)*hrhol(m,n+1))
          den=ssq(n)*(jrho2(m,n)+gamma(m,n)*hrho2(m,n))
          xjc(m,n)=num*xjc(m,n+1)/den
          xhc(m,n)=gamma(m,n)*xjc(m,n)
          xjs(m,n)=num*xjs(m,n+1)/den
          xhs(m,n)=gamma(m,n)*xjs(m,n)
        end do
        num=ssq(2)*(jrhol(m,2)+gamma(m,2)*hrhol(m,2))
        den=ssq(1)*jrho2(m,1)
        xjc(m,1)=num*xjc(m,2)/den
        xjs(m,1)=num*xjs(m,2)/den
      end do
```

```
  go to 2
end if
if(nt .eq. 2 .and. nrslab .gt. 3)then
  do m=0,mmax-1
    a4(1,1)=-ssq(1)*jrho2(m,1)
    a4(1,2)=ssq(2)*jrho1(m,2)
    a4(1,3)=ssq(2)*hrho1(m,2)
    a4(1,4)=0.
    a4(2,1)=-s(1)*jprho2(m,1)
    a4(2,2)=s(2)*jprho1(m,2)
    a4(2,3)=s(2)*hprho1(m,2)
    a4(2,4)=0.
    a4(3,1)=0.
    a4(3,2)=ssq(2)*jrho2(m,2)
    a4(3,3)=ssq(2)*hrho2(m,2)
    a4(3,4)=-ssq(3)*(gamma1(m,3)*jrho1(m,3)+hrho1(m,3))
    a4(4,1)=0.
    a4(4,2)=s(2)*jprho2(m,2)
    a4(4,3)=s(2)*hprho2(m,2)
    a4(4,4)=-s(3)*(gamma1(m,3)*jprho1(m,3)+hprho1(m,3))
    a4sav(1,1)=a4(1,1)
    a4sav(1,2)=a4(1,2)
    a4sav(1,3)=a4(1,3)
    a4sav(1,4)=a4(1,4)
    a4sav(2,1)=a4(2,1)
    a4sav(2,2)=a4(2,2)
    a4sav(2,3)=a4(2,3)
    a4sav(2,4)=a4(2,4)
    a4sav(3,1)=a4(3,1)
    a4sav(3,2)=a4(3,2)
    a4sav(3,3)=a4(3,3)
    a4sav(3,4)=a4(3,4)
    a4sav(4,1)=a4(4,1)
    a4sav(4,2)=a4(4,2)
    a4sav(4,3)=a4(4,3)
    a4sav(4,4)=a4(4,4)
    b4(1)=ssq(2)*ampc*hpr0(m)*jrho1(m,2)
    b4(2)=s(2)*ampc*hpr0(m)*jprho1(m,2)
    b4(3)=ssq(2)*ampc*jpr0(m)*hrho2(m,2)
    b4(4)=s(2)*ampc*jpr0(m)*hprho2(m,2)
    if(m .eq. 0)then
      b4(1)=.5*b4(1)
      b4(2)=.5*b4(2)
      b4(3)=.5*b4(3)
      b4(4)=.5*b4(4)
    end if
    call clineq(a4,b4,x4,4,4,0,err)
    xjc(m,1)=x4(1)
    xjc(m,2)=x4(2)
    xhc(m,2)=x4(3)
    xhc(m,3)=x4(4)
    b4(1)=-ssq(2)*amps*m*gr0(m)*jrho1(m,2)
    b4(2)=-s(2)*amps*m*gr0(m)*jprho1(m,2)
    b4(3)=-ssq(2)*amps*m*lr0(m)*hrho2(m,2)
    b4(4)=-s(2)*amps*m*lr0(m)*hprho2(m,2)
    call clineq(a4sav,b4,x4,4,4,0,err)
    xjs(m,1)=x4(1)
    xjs(m,2)=x4(2)
```

```
        xhs(m,2)=x4(3)
        xhs(m,3)=x4(4)
        xjc(m,3)=gammal(m,3)*xhc(m,3)
        xjs(m,3)=gammal(m,3)*xhs(m,3)
        do n=4,nrslab-1
          num=ssq(n-1)*(gammal(m,n-1)*jrho2(m,n-1)+hrho2(m,n-1))
          den=ssq(n)*(gammal(m,n)*jrhol(m,n)+hrhol(m,n))
          xhc(m,n)=num*xhc(m,n-1)/den
          xjc(m,n)=gammal(m,n)*xhc(m,n)
          xhs(m,n)=num*xhs(m,n-1)/den
          xjs(m,n)=gammal(m,n)*xhs(m,n)
        end do
        num=ssq(nrslab-1)*(gammal(m,nrslab-1)*jrho2(m,nrslab-1)+
     $       hrho2(m,nrslab-1))
        den=ssq(nrslab)*hrhol(m,nrslab)
        xhc(m,nrslab)=num*xhc(m,nrslab-1)/den
        xhs(m,nrslab)=num*xhs(m,nrslab-1)/den
      end do
      go to 2
    end if
    if(nt .eq. nrslab-1 .and. nrslab .gt. 3)then
      do m=0,mmax-1
        a4(1,1)=-ssq(nrslab-2)*(jrho2(m,nrslab-2)+gamma(m,nrslab-2)*
     $           hrho2(m,nrslab-2))
        a4(1,2)=ssq(nrslab-1)*jrhol(m,nrslab-1)
        a4(1,3)=ssq(nrslab-1)*hrhol(m,nrslab-1)
        a4(1,4)=0.
        a4(2,1)=-s(nrslab-2)*(jprho2(m,nrslab-2)+gamma(m,nrslab-2)*
     $           hprho2(m,nrslab-2))
        a4(2,2)=s(nrslab-1)*jprhol(m,nrslab-1)
        a4(2,3)=s(nrslab-1)*hprhol(m,nrslab-1)
        a4(2,4)=0.
        a4(3,1)=0.
        a4(3,2)=ssq(nrslab-1)*jrho2(m,nrslab-1)
        a4(3,3)=ssq(nrslab-1)*hrho2(m,nrslab-1)
        a4(3,4)=-ssq(nrslab)*hrhol(m,nrslab)
        a4(4,1)=0.
        a4(4,2)=s(nrslab-1)*jprho2(m,nrslab-1)
        a4(4,3)=s(nrslab-1)*hprho2(m,nrslab-1)
        a4(4,4)=-s(nrslab)*hprhol(m,nrslab)
        a4sav(1,1)=a4(1,1)
        a4sav(1,2)=a4(1,2)
        a4sav(1,3)=a4(1,3)
        a4sav(1,4)=a4(1,4)
        a4sav(2,1)=a4(2,1)
        a4sav(2,2)=a4(2,2)
        a4sav(2,3)=a4(2,3)
        a4sav(2,4)=a4(2,4)
        a4sav(3,1)=a4(3,1)
        a4sav(3,2)=a4(3,2)
        a4sav(3,3)=a4(3,3)
        a4sav(3,4)=a4(3,4)
        a4sav(4,1)=a4(4,1)
        a4sav(4,2)=a4(4,2)
        a4sav(4,3)=a4(4,3)
        a4sav(4,4)=a4(4,4)
        b4(1)=ssq(nrslab-1)*ampc*hpr0(m)*jrhol(m,nrslab-1)
        b4(2)=s(nrslab-1)*ampc*hpr0(m)*jprhol(m,nrslab-1)
```

```
            b4(3)=ssq(nrslab-1)*ampc*jpr0(m)*hrho2(m,nrslab-1)
            b4(4)=s(nrslab-1)*ampc*jpr0(m)*hprho2(m,nrslab-1)
            if(m .eq. 0)then
              b4(1)=.5*b4(1)
              b4(2)=.5*b4(2)
              b4(3)=.5*b4(3)
              b4(4)=.5*b4(4)
            end if
            call clineq(a4,b4,x4,4,4,0,err)
            xjc(m,nrslab-2)=x4(1)
            xjc(m,nrslab-1)=x4(2)
            xhc(m,nrslab-1)=x4(3)
            xhc(m,nrslab)=x4(4)
            b4(1)=-ssq(nrslab-1)*amps*m*gr0(m)*jrhol(m,nrslab-1)
            b4(2)=-s(nrslab-1)*amps*m*gr0(m)*jprhol(m,nrslab-1)
            b4(3)=-ssq(nrslab-1)*amps*m*lr0(m)*hrho2(m,nrslab-1)
            b4(4)=-s(nrslab-1)*amps*m*lr0(m)*hprho2(m,nrslab-1)
            call clineq(a4sav,b4,x4,4,4,0,err)
            xjs(m,nrslab-2)=x4(1)
            xjs(m,nrslab-1)=x4(2)
            xhs(m,nrslab-1)=x4(3)
            xhs(m,nrslab)=x4(4)
            xhc(m,nrslab-2)=gamma(m,nrslab-2)*xjc(m,nrslab-2)
            xhs(m,nrslab-2)=gamma(m,nrslab-2)*xjs(m,nrslab-2)
            do n=nrslab-3,2,-1
              num=ssq(n+1)*(jrhol(m,n+1)+gamma(m,n+1)*hrhol(m,n+1))
              den=ssq(n)*(jrho2(m,n)+gamma(m,n)*hrho2(m,n))
              xjc(m,n)=num*xjc(m,n+1)/den
              xhc(m,n)=gamma(m,n)*xjc(m,n)
              xjs(m,n)=num*xjs(m,n+1)/den
              xhs(m,n)=gamma(m,n)*xjs(m,n)
            end do
            num=ssq(2)*(jrhol(m,2)+gamma(m,2)*hrhol(m,2))
            den=ssq(1)*jrho2(m,1)
            xjc(m,1)=num*xjc(m,2)/den
            xjs(m,1)=num*xjs(m,2)/den
          end do
          go to 2
        end if
c nt>4,nt .ne. 1,2,nrslab-1,nrslab
        do m=0,mmax-1
          a4(1,1)=-ssq(nt-1)*(jrho2(m,nt-1)+gamma(m,nt-1)*hrho2(m,nt-1))
          a4(1,2)=ssq(nt)*jrhol(m,nt)
          a4(1,3)=ssq(nt)*hrhol(m,nt)
          a4(1,4)=0.
          a4(2,1)=-s(nt-1)*(jprho2(m,nt-1)+gamma(m,nt-1)*hprho2(m,nt-1))
          a4(2,2)=s(nt)*jprhol(m,nt)
          a4(2,3)=s(nt)*hprhol(m,nt)
          a4(2,4)=0.
          a4(3,1)=0.
          a4(3,2)=ssq(nt)*jrho2(m,nt)
          a4(3,3)=ssq(nt)*hrho2(m,nt)
          a4(3,4)=-ssq(nt+1)*(gammal(m,nt+1)*jrhol(m,nt+1)+hrhol(m,nt+1))
          a4(4,1)=0.
          a4(4,2)=s(nt)*jprho2(m,nt)
          a4(4,3)=s(nt)*hprho2(m,nt)
          a4(4,4)=-s(nt+1)*(gammal(m,nt+1)*jprhol(m,nt+1)+hprhol(m,nt+1))
           a4sav(1,1)=a4(1,1)
```

```
      a4sav(1,2)=a4(1,2)
      a4sav(1,3)=a4(1,3)
      a4sav(1,4)=a4(1,4)
      a4sav(2,1)=a4(2,1)
      a4sav(2,2)=a4(2,2)
      a4sav(2,3)=a4(2,3)
      a4sav(2,4)=a4(2,4)
      a4sav(3,1)=a4(3,1)
      a4sav(3,2)=a4(3,2)
      a4sav(3,3)=a4(3,3)
      a4sav(3,4)=a4(3,4)
      a4sav(4,1)=a4(4,1)
      a4sav(4,2)=a4(4,2)
      a4sav(4,3)=a4(4,3)
      a4sav(4,4)=a4(4,4)
     b4(1)=ssq(nt)*ampc*hpr0(m)*jrho1(m,nt)
     b4(2)=s(nt)*ampc*hpr0(m)*jprho1(m,nt)
     b4(3)=ssq(nt)*ampc*jpr0(m)*hrho2(m,nt)
     b4(4)=s(nt)*ampc*jpr0(m)*hprho2(m,nt)
     if(m .eq. 0)then
       b4(1)=.5*b4(1)
       b4(2)=.5*b4(2)
       b4(3)=.5*b4(3)
       b4(4)=.5*b4(4)
     end if
     call clineq(a4,b4,x4,4,4,0,err)
     xjc(m,nt-1)=x4(1)
     xjc(m,nt)=x4(2)
     xhc(m,nt)=x4(3)
     xhc(m,nt+1)=x4(4)
     b4(1)=-ssq(nt)*amps*m*gr0(m)*jrho1(m,nt)
     b4(2)=-s(nt)*amps*m*gr0(m)*jprho1(m,nt)
     b4(3)=-ssq(nt)*amps*m*lr0(m)*hrho2(m,nt)
     b4(4)=-s(nt)*amps*m*lr0(m)*hprho2(m,nt)
     call clineq(a4sav,b4,x4,4,4,0,err)
     xjs(m,nt-1)=x4(1)
     xjs(m,nt)=x4(2)
     xhs(m,nt)=x4(3)
     xhs(m,nt+1)=x4(4)
     xhc(m,nt-1)=gamma(m,nt-1)*xjc(m,nt-1)
     xjc(m,nt+1)=gamma1(m,nt+1)*xhc(m,nt+1)
     xhs(m,nt-1)=gamma(m,nt-1)*xjs(m,nt-1)
     xjs(m,nt+1)=gamma1(m,nt+1)*xhs(m,nt+1)
     do n=nt+2,nrslab-1
       num=ssq(n-1)*(gamma1(m,n-1)*jrho2(m,n-1)+hrho2(m,n-1))
       den=ssq(n)*(gamma1(m,n)*jrho1(m,n)+hrho1(m,n))
       xhc(m,n)=num*xhc(m,n-1)/den
       xjc(m,n)=gamma1(m,n)*xhc(m,n)
       xhs(m,n)=num*xhs(m,n-1)/den
       xjs(m,n)=gamma1(m,n)*xhs(m,n)
     end do
     num=ssq(nrslab-1)*(gamma1(m,nrslab-1)*jrho2(m,nrslab-1)+
$        hrho2(m,nrslab-1))
     den= ssq(nrslab)*hrho1(m,nrslab)
     xhc(m,nrslab)=num*xhc(m,nrslab-1)/den
     xhs(m,nrslab)=num*xhs(m,nrslab-1)/den
     do n=nt-2,2,-1
       num=ssq(n+1)*(jrho1(m,n+1)+gamma(m,n+1)*hrho1(m,n+1))
```

```
          den=ssq(n)*(jrho2(m,n)+gamma(m,n)*hrho2(m,n))
          xjc(m,n)=num*xjc(m,n+1)/den
          xhc(m,n)=gamma(m,n)*xjc(m,n)
          xjs(m,n)=num*xjs(m,n+1)/den
          xhs(m,n)=gamma(m,n)*xjs(m,n)
        end do
        num=ssq(2)*(jrhol(m,2)+gamma(m,2)*hrhol(m,2))
        den=ssq(1)*jrho2(m,1)
        xjc(m,1)=num*xjc(m,2)/den
        xjs(m,1)=num*xjs(m,2)/den
      end do
2     continue
      ntsav=nt
      return
      end
```

```
      subroutine mlflds
      implicit complex*16(a-h,o-z)
      complex*16 ks,im,jprhor,jrhor,jprhot,jrhot,jpr0,jr0,lr0
      real*8 rmin,r,rsav,alpha,cosalp,sinalp,smin,x0,xyint,y0,r0,rinc,
     $rmax,sinc,smax,ctheta,rhorcv,pi,phi,stheta,rho,cmphi,smphi,chi,
     $cchi,schi,ctmd1,ctmd2,delta1,delta2,stmd1,stmd2,theta,scoord,
     $ezdb,ezang,hydb,hyang,hrdb,hrang,skip
      real*4 xplot,yplot1,yplot2,yplot3,yplot4,yplot5,yplot6
      parameter(mmax=120)
      parameter(mxslab=66)
      parameter(mxplot=302)
      dimension rho(mxslab),ks(mxslab),s(mxslab),ssq(mxslab),
     $hprhor(0:mmax),hrhor(0:mmax),jprhor(0:mmax),jrhor(0:mmax),
     $xjs(0:mmax,mxslab),xjc(0:mmax,mxslab),xhs(0:mmax,mxslab),
     $xhc(0:mmax,mxslab),jprhot(0:mmax),jrhot(0:mmax),hprhot(0:mmax),
     $hrhot(0:mmax),jr0(0:mmax),jpr0(0:mmax),hr0(0:mmax),hpr0(0:mmax),
     $bjr0(0:mmax),bjpr0(0:mmax),bhr0(0:mmax),bhpr0(0:mmax),
     $bgr0(0:mmax),blr0(0:mmax),
     $gr0(0:mmax),lr0(0:mmax),xplot(mxplot),yplot1(mxplot),
     $yplot2(mxplot),yplot3(mxplot),yplot4(mxplot),yplot5(mxplot),
     $yplot6(mxplot)
      common/one/ampc,amps,ks,nrslab,r0,rho,s,ssq
      common/three/alpha,ampprc,ampprs,c1,c2,cosalp,delta1,delta2,
     $iflag,rinc,rmax,rmin,sinalp,sinc,smax,smin,xyint,theta,x0,y0,skip
      common/four/xhc,xhs,xjc,xjs
      common/five/gr0,hpr0,hr0,jpr0,jr0,lr0
      data pi/3.1415926536/
      data im/(0.,1.)/
      r=rmin
      if(r .eq. 0.)r=1.d-4
      rsav=0.
      rtsav=nt
      scoord=smin
      stheta=sin(theta)
      ctheta=cos(theta)
      if(iflag .eq. 0)then
        numpts=(rmax-rmin)/rinc+1
        write(40) x0,y0,numpts
      else
        numpts=(smax-smin)/sinc+1
        write(40) r,numpts
      end if
      do num=1,numpts
        rhorcv=sqrt(r*r+r0*r0-2.*r*r0*ctheta)
        if(rhorcv .lt. 1.)rhorcv=1.
        if(r0*r0+rhorcv*rhorcv-r*r .eq. 0.)then
          phi=pi/2.
          if(stheta .lt. 0.)phi=-phi
        else
          phi=atan2(2.*r*r0*stheta,r0*r0+rhorcv*rhorcv-r*r)
        end if
        nr=nrslab
        do n=1,nrslab-1
           if(rhorcv .le. rho(n))then
             nr=n
             go to 1
```

```
            end if
         end do
 1       continue
         nt=nrslab
         do n=1,nrslab-1
           if(r0 .le. rho(n))then
             nt=n
             go to 2
           end if
         end do
 2       continue
         if(abs(r-r0) .lt. 150.)go to 3
         if(abs(rhorcv-r0) .lt. skip)go to 3
         argr0=ks(nr)*r0
         call hjfunc(argr0,bjr0,bjpr0,bhr0,bhpr0)
         do m=0,mmax-1
           blr0(m)=bjr0(m)/argr0
           bgr0(m)=bhr0(m)/argr0
         end do
         if(r .ne. rsav)then
           arg=ks(nrslab)*r
           call cbesjy(arg,0,bj,by,0,0)
           h0=bj-im*by
           call cbesjy(arg,1,bj,by,0,0)
           h1=bj-im*by
           call cbesjy(arg,2,bj,by,0,0)
           h2=bj-im*by
           hp1=0.5*(h0-h2)
           ezamb=ssq(nrslab)*ampprc*h1
           hyamb=-im*s(nrslab)*ampprc*hp1
           hramb=im*s(nrslab)*ampprs*h1/arg
         end if
         if(nt .ne. ntsav .or. r .ne. rsav)then
           arg=ks(nt)*r
           call cbesjy(arg,0,bj,by,0,0)
           h0=bj-im*by
           call cbesjy(arg,1,bj,by,0,0)
           h1=bj-im*by
           call cbesjy(arg,2,bj,by,0,0)
           h2=bj-im*by
           hp1=0.5*(h0-h2)
           eztst=ssq(nt)*ampprc*h1
           hytst=-im*s(nt)*ampprc*hp1
           hrtst=im*s(nt)*ampprs*h1/arg
         end if
           ntsav=nt
c         if(nr .eq. nt)then
           arg=ks(nr)*r
           call cbesjy(arg,0,bj,by,0,0)
           h0=bj-im*by
           call cbesjy(arg,1,bj,by,0,0)
           h1=bj-im*by
           call cbesjy(arg,2,bj,by,0,0)
           h2=bj-im*by
           hp1=0.5*(h0-h2)
           ezprmy=ssq(nr)*ampprc*h1
           hyprmy=-im*s(nr)*ampprc*hp1
           hrprmy=im*s(nr)*ampprs*h1/arg
```

```
        rsav=r
c      end if
      arg=ks(nt)*rhorcv
      ezsum=0.
      hosum=0.
      hrsum=0.
      call hjfunc(arg,jrhot,jprhot,hrhot,hprhot)
      do m=mmax-1,0,-1
        cmphi=cos(m*phi)
        smphi=sin(m*phi)
        if(rhorcv .lt. r0)then
          if(m .eq. 0)then
            ezsum=ezsum-ampc*.5*ssq(nt)*hpr0(m)*jrhot(m)*cmphi
            hosum=hosum+ampc*.5*im*s(nt)*hpr0(m)*jprhot(m)*cmphi
          else
            ezsum=ezsum-ampc*ssq(nt)*hpr0(m)*jrhot(m)*cmphi
     $            +amps*ssq(nt)*m*gr0(m)*jrhot(m)*smphi
            hosum=hosum+ampc*im*s(nt)*hpr0(m)*jprhot(m)*cmphi
     $            -amps*im*s(nt)*m*gr0(m)*jprhot(m)*smphi
            hrsum=hrsum+ampc*im*s(nt)*m*hpr0(m)*jrhot(m)*smphi/arg
     $            +amps*im*s(nt)*m*m*gr0(m)*jrhot(m)*cmphi/arg
          end if
        else
          if(m .eq. 0) then
            ezsum=ezsum-ampc*.5*ssq(nt)*jpr0(m)*hrhot(m)*cmphi
            hosum=hosum+ampc*.5*im*s(nt)*jpr0(m)*hprhot(m)*cmphi
          else
            ezsum=ezsum-ampc*ssq(nt)*jpr0(m)*hrhot(m)*cmphi
     $            +amps*ssq(nt)*m*lr0(m)*hrhot(m)*smphi
            hosum=hosum+ampc*im*s(nt)*jpr0(m)*hprhot(m)*cmphi
     $            -amps*im*s(nt)*m*lr0(m)*hprhot(m)*smphi
            hrsum=hrsum+ampc*im*s(nt)*m*jpr0(m)*hrhot(m)*smphi/arg
     $            +amps*im*s(nt)*m*m*lr0(m)*hrhot(m)*cmphi/arg
          end if
        end if
      end do
      chi=pi-theta-phi
      cchi=cos(chi)
      schi=sin(chi)
      hysum=hosum*cchi-hrsum*schi
      hrsum=hosum*schi+hrsum*cchi
      write(44,*)'nt=',nt,'nr=',nr
      write(44,*)'chi=',chi,'theta=',theta,'phi=',phi
      write(44,*)'r=',r,'r0=',r0,'rhorcv=',rhorcv
      write(44,*)'x0=',x0,'y0=',y0
      if(zabs(eztst-ezsum) .gt. .005 .or. zabs(hytst-hysum) .gt. .005
     $   .or. zabs(hrtst-hrsum) .gt. .005)then
        write(44,*)'convergence warning'
        write(44,*)'eztst=',eztst,'ezps=',ezsum
        write(44,*)'hytst=',hytst,'hyps=',hysum
        write(44,*)'hrtst=',hrtst,'hrps=',hrsum
      end if
      arg=ks(nr)*rhorcv
      call hjfunc(arg,jrhor,jprhor,hrhor,hprhor)
      ezs=0.
      hphis=0.
      hrhos=0.
      do m=mmax-1,0,-1
```

```
c           write(44,*)'m=',m,'nr=',nr,'xhc=',xhc(m,nr),'xhs=',xhs(m,nr)
          cmphi=cos(m*phi)
          smphi=sin(m*phi)
        if(nt .eq. nr)then
          if(nr .eq. 1)then
                tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
                templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
                ezs=ezs+ssq(nr)*tempj*jrhor(m)
                hphis=hphis-im*s(nr)*tempj*jprhor(m)
                hrhos=hrhos+im*s(nr)*templj*jrhor(m)/arg
          else
            if(nr .eq. nrslab)then
                  temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
                  templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
                  ezs=ezs+ssq(nr)*temph*hrhor(m)
                  hphis=hphis-im*s(nr)*temph*hprhor(m)
                  hrhos=hrhos+im*s(nr)*templh*hrhor(m)/arg
            else
                  tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
                  temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
                  templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
                  templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
                  ezs=ezs+ssq(nr)*(tempj*jrhor(m)+temph*hrhor(m))
                  hphis=hphis-im*s(nr)*
     $                  (tempj*jprhor(m)+temph*hprhor(m))
                  hrhos=hrhos+im*s(nr)*
     $                  (templj*jrhor(m)+templh*hrhor(m))/arg
            end if
          end if
                  go to 10
          end if
          if(nr .eq. 1)then
            if(rhorcv .lt. r0)then
              if(m .eq. 0)then
                tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
                templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
                ezs=ezs+ssq(nr)*tempj*jrhor(m)
     $              +ampc*.5*ssq(nr)*bhpr0(m)*jrhor(m)*cmphi
                hphis=hphis-im*s(nr)*tempj*jprhor(m)
     $              -ampc*.5*im*s(nr)*bhpr0(m)*jprhor(m)*cmphi
                hrhos=hrhos+im*s(nr)*templj*jrhor(m)/arg
              else
                tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
                templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
                ezs=ezs+ssq(nr)*tempj*jrhor(m)
     $              +ampc*ssq(nr)*bhpr0(m)*jrhor(m)*cmphi
     $              -amps*ssq(nr)*m*bgr0(m)*jrhor(m)*smphi
                hphis=hphis-im*s(nr)*tempj*jprhor(m)
     $              -ampc*im*s(nr)*bhpr0(m)*jprhor(m)*cmphi
     $              +amps*im*s(nr)*m*bgr0(m)*jprhor(m)*smphi
                hrhos=hrhos+im*s(nr)*templj*jrhor(m)/arg
     $              -ampc*im*s(nr)*m*bhpr0(m)*jrhor(m)*smphi/arg
     $              -amps*im*s(nr)*m*m*bgr0(m)*jrhor(m)*cmphi/arg
              end if
            else
              if(m .eq. 0)then
                tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
                templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
```

```
            ezs=ezs+ssq(nr)*tempj*jrhor(m)
     $          +ampc*.5*ssq(nr)*bjpr0(m)*hrhor(m)*cmphi
            hphis=hphis-im*s(nr)*tempj*jprhor(m)
     $            -ampc*.5*im*s(nr)*bjpr0(m)*hprhor(m)*cmphi
            hrhos=hrhos+im*s(nr)*templj*jrhor(m)/arg
          else
            tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
            templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
            ezs=ezs+ssq(nr)*tempj*jrhor(m)
     $          +ampc*ssq(nr)*bjpr0(m)*hrhor(m)*cmphi
     $          -amps*ssq(nr)*m*blr0(m)*hrhor(m)*smphi
            hphis=hphis-im*s(nr)*tempj*jprhor(m)
     $            -ampc*im*s(nr)*bjpr0(m)*hprhor(m)*cmphi
     $            +amps*im*s(nr)*m*blr0(m)*hprhor(m)*smphi
            hrhos=hrhos+im*s(nr)*templj*jrhor(m)/arg
     $            -ampc*im*s(nr)*m*bjpr0(m)*hrhor(m)*smphi/arg
     $            -amps*im*s(nr)*m*m*blr0(m)*hrhor(m)*cmphi/arg
          end if
        end if
      else
        if(nr .eq. nrslab)then
          if(rhorcv .lt. r0)then
            if(m .eq. 0)then
              temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
              templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
              ezs=ezs+ssq(nr)*temph*hrhor(m)
     $            +ampc*.5*ssq(nr)*bhpr0(m)*jrhor(m)*cmphi
              hphis=hphis-im*s(nr)*temph*hprhor(m)
     $              -ampc*.5*im*s(nr)*bhpr0(m)*jprhor(m)*cmphi
              hrhos=hrhos+im*s(nr)*templh*hrhor(m)/arg
            else
              temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
              templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
              ezs=ezs+ssq(nr)*temph*hrhor(m)
     $            +ampc*ssq(nr)*bhpr0(m)*jrhor(m)*cmphi
     $            -amps*ssq(nr)*m*bgr0(m)*jrhor(m)*smphi
              hphis=hphis-im*s(nr)*temph*hprhor(m)
     $              -ampc*im*s(nr)*bhpr0(m)*jprhor(m)*cmphi
     $              +amps*im*s(nr)*m*bgr0(m)*jprhor(m)*smphi
              hrhos=hrhos+im*s(nr)*templh*hrhor(m)/arg
     $              -ampc*im*s(nr)*m*bhpr0(m)*jrhor(m)*smphi/arg
     $              -amps*im*s(nr)*m*m*bgr0(m)*jrhor(m)*cmphi/arg
            end if
          else
            if(m .eq. 0)then
              temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
              templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
              ezs=ezs+ssq(nr)*temph*hrhor(m)
     $            +ampc*.5*ssq(nr)*bjpr0(m)*hrhor(m)*cmphi
              hphis=hphis-im*s(nr)*temph*hprhor(m)
     $              -ampc*.5*im*s(nr)*bjpr0(m)*hprhor(m)*cmphi
              hrhos=hrhos+im*s(nr)*templh*hrhor(m)/arg
            else
              temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
              templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
              ezs=ezs+ssq(nr)*temph*hrhor(m)
     $            +ampc*ssq(nr)*bjpr0(m)*hrhor(m)*cmphi
     $            -amps*ssq(nr)*m*blr0(m)*hrhor(m)*smphi
```

```
            hphis=hphis-im*s(nr)*temph*hprhor(m)
     $            -ampc*im*s(nr)*bjpr0(m)*hprhor(m)*cmphi
     $            +amps*im*s(nr)*m*blr0(m)*hprhor(m)*smphi
            hrhos=hrhos+im*s(nr)*templh*hrhor(m)/arg
     $            -ampc*im*s(nr)*m*bjpr0(m)*hrhor(m)*smphi/arg
     $            -amps*im*s(nr)*m*m*blr0(m)*hrhor(m)*cmphi/arg
           end if
        end if
      else
        if(rhorcv .lt. r0)then
          if(m .eq. 0)then
            tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
            temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
            templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
            templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
            ezs=ezs+ssq(nr)*(tempj*jrhor(m)+temph*hrhor(m))
     $          +ampc*.5*ssq(nr)*bhpr0(m)*jrhor(m)*cmphi
            hphis=hphis-im*s(nr)*
     $            (tempj*jprhor(m)+temph*hprhor(m))
     $            -ampc*.5*im*s(nr)*bhpr0(m)*jprhor(m)*cmphi
            hrhos=hrhos+im*s(nr)*
     $             (templj*jrhor(m)+templh*hrhor(m))/arg
          else
            tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
            temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
            templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
            templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
            ezs=ezs+ssq(nr)*(tempj*jrhor(m)+temph*hrhor(m))
     $          +ampc*ssq(nr)*bhpr0(m)*jrhor(m)*cmphi
     $          -amps*ssq(nr)*m*bgr0(m)*jrhor(m)*smphi
            hphis=hphis-im*s(nr)*
     $            (tempj*jprhor(m)+temph*hprhor(m))
     $            -ampc*im*s(nr)*bhpr0(m)*jprhor(m)*cmphi
     $            +amps*im*s(nr)*m*bgr0(m)*jprhor(m)*smphi
            hrhos=hrhos+im*s(nr)*
     $            (templj*jrhor(m)+templh*hrhor(m))/arg
     $            -ampc*im*s(nr)*m*bhpr0(m)*jrhor(m)*smphi/arg
     $            -amps*im*s(nr)*m*m*bgr0(m)*jrhor(m)*cmphi/arg
          end if
        else
          if(m .eq. 0)then
            tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
            temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
            templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
            templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
            ezs=ezs+ssq(nr)*(tempj*jrhor(m)+temph*hrhor(m))
     $          +ampc*.5*ssq(nr)*bjpr0(m)*hrhor(m)*cmphi
            hphis=hphis-im*s(nr)*
     $            (tempj*jprhor(m)+temph*hprhor(m))
     $            -ampc*.5*im*s(nr)*bjpr0(m)*hprhor(m)*cmphi
            hrhos=hrhos+im*s(nr)*
     $            (templj*jrhor(m)+templh*hrhor(m))/arg
          else
            tempj=(xjc(m,nr)*cmphi+xjs(m,nr)*smphi)
            temph=(xhc(m,nr)*cmphi+xhs(m,nr)*smphi)
            templj=m*(-xjc(m,nr)*smphi+xjs(m,nr)*cmphi)
            templh=m*(-xhc(m,nr)*smphi+xhs(m,nr)*cmphi)
            ezs=ezs+ssq(nr)*(tempj*jrhor(m)+temph*hrhor(m))
```

```
     $                +ampc*ssq(nr)*bjpr0(m)*hrhor(m)*cmphi
     $                -amps*ssq(nr)*m*blr0(m)*hrhor(m)*smphi
                  hphis=hphis-im*s(nr)*
     $                  (tempj*jprhor(m)+temph*hprhor(m))
     $                  -ampc*im*s(nr)*bjpr0(m)*hprhor(m)*cmphi
     $                  +amps*im*s(nr)*m*blr0(m)*hprhor(m)*smphi
                  hrhos=hrhos+im*s(nr)*
     $                  (templj*jrhor(m)+templh*hrhor(m))/arg
     $                  -ampc*im*s(nr)*m*bjpr0(m)*hrhor(m)*smphi/arg
     $                  -amps*im*s(nr)*m*m*blr0(m)*hrhor(m)*cmphi/arg
                end if
              end if
            end if
          end if
10      continue
        end do
        chi=pi-theta-phi
        cchi=cos(chi)
        schi=sin(chi)
          ez=ezprmy+ezs
          hy=hyprmy+hphis*cchi-hrhos*schi
          hr=hrprmy+hphis*schi+hrhos*cchi
        write(44,*)'scoord=',scoord
c        write(44,*)'ezamb=',ezamb
        if(zabs(ampprc) .ne. 0.)then
          qez=zlog(ez/ezamb)
          qhy=zlog(hy/hyamb)
        end if
        if(zabs(ampprs) .ne. 0.)qhr=zlog(hr/hramb)
        ezdb=8.6858896*qez
        hydb=8.6858896*qhy
        hrdb=8.6858896*qhr
        ezang=-im*qez*57.295779
        hyang=-im*qhy*57.295779
        hrang=-im*qhr*57.295779
        if(ezang .gt. 180.)ezang=ezang-360.
        if(ezang .lt. -180.)ezang=ezang+360.
        ezang=-ezang
        if(hyang .gt. 180.)hyang=hyang-360.
        if(hyang .lt. -180.)hyang=hyang+360.
        hyang=-hyang
        if(hrang .gt. 180.)hrang=hrang-360.
        if(hrang .lt. -180.)hrang=hrang+360.
        hrang=-hrang
        yplot1(num)=sngl(ezdb)
        yplot2(num)=sngl(hydb)
        yplot3(num)=sngl(hrdb)
        yplot4(num)=sngl(ezang)
        yplot5(num)=sngl(hyang)
        yplot6(num)=sngl(hrang)
        if(iflag .eq. 0)then
          xplot(num)=sngl(r)
        else
          xplot(num)=sngl(scoord)
        end if
        write(44,*)'ez/ezu(db)=',ezdb
        write(44,*)'ez/^zu(deg)=',ezang
        write(44,*)'hy/hyu(db)=',hydb
```

```
      write(44,*)'hy/hyu(deg)=',hyang
      write(44,*)'hr/hru(db)=',hrdb
      write(44,*)'hr/hru(deg)=',hrang
      write(40) xplot(num),yplot1(num),yplot2(num),yplot3(num),
     $                  yplot4(num),yplot5(num),yplot6(num)
3     continue
      if(iflag .eq. 0)then
        r=r+rinc
        if(r .eq. 0.)r=1.d-4
        if(r .gt. rmax+.01)go to 5
      else
        if(alpha .eq. 0.)then
          x0=x0+sinc
          y0=xyint
          scoord=scoord+sinc
        else
          x0=x0+sinc*cosalp
          y0=y0+sinc*sinalp
          scoord=scoord+sinc
        end if
        if(scoord .gt. smax+.01)go to 5
        r0=sqrt(x0**2+y0**2)
        if(r0 .lt. 1.)then
          if(x0 .lt. 0.)then
            x0=-1.
            r0=sqrt(x0**2+y0**2)
          else
            x0=1.
            r0=sqrt(x0**2+y0**2)
          end if
          write(44,*)'r0=',r0
          go to 4
        end if
4       if(x0 .eq. 0.)then
          theta=pi/2.
          if(y0 .lt. 0.)theta=-theta
        else
          theta=atan2(y0,x0)
        end if
        ctmd1=cos(theta-delta1)
        ctmd2=cos(theta-delta2)
        stmd1=sin(theta-delta1)
        stmd2=sin(theta-delta2)
        stheta=sin(theta)
        ctheta=cos(theta)
        ampc=2.*ctmd1*c1+2.*ctmd2*c2
        amps=2.*stmd1*c1+2.*stmd2*c2
        call mlcoef
      end if
    end do
5   return
    end
```

```
      SUBROUTINE CBESJY(Z,K,BJ,BY,KIND,NPRINT)
      IMPLICIT COMPLEX*16 (A-H,O-Z)
      REAL*8 DGMSUM,DG1,DG2,PI,RRT,EULER,KFAC,KM1FAC
      data PI/3.14159265358979D0/,EULER/0.577215664901533D0/
C
      IF(ZABS(Z) .NE. 0.D0) GO TO 7
      BJ=(0.0D0,0.0D0)
      IF(K .EQ. 0) BJ=(1.0D0,0.0D0)
      IF(KIND .NE. 1 .AND. K .EQ. 0) PRINT 400
400   FORMAT(1H0,'*** Y NOT CALCULATED FOR ARGUMENT OF MAGNITUDE 0'//)
      RETURN
7     IF(ZABS(Z) .LT. 13.D0) GO TO 10
C
C
C     ASSYMPTOTIC EXPANSION
C
      RHO=8.*Z
      MU=4*K**2
      RT=ZSQRT(2./(PI*Z))
      RRT=RT
      IF(RRT .LT. 0.0D0) RT=-RT
      P=0.
C  DO LOOP FOR CALCULATING P
      DO 1 N=1,30
      M=N-1
      MM=2*M
      IF(N .EQ. 1) GO TO 2
      TERM=(-1)*(MU-(2*MM-3)**2)*(MU-(2*MM-1)**2)/(MM*(MM-1)*RHO**2)*
     $      TEPM
      IF(NPRINT .EQ. 1) PRINT 100,TERM
100   FORMAT(' TERM=',2D30.15)
      P=P+TERM
      IF(ZABS(TERM) .GE. ZABS(TERMS) .OR. ZABS(TERM) .LE. 1.D-17)
     $   GO TO 3
      TERMS=TERM
      GO TO 1
2     TERM=1.D0
      P=TERM
      IF(NPRINT .EQ. 1) PRINT 100,TERM
      TERMS=TERM
1     CONTINUE
3     CONTINUE
      IF(NPRINT .EQ. 1) PRINT 200,P
200   FORMAT(1H0,'P=',2D30.15)
      Q=0.
C
C  DO LOOP FOR CALCULATING Q
      DO 4 N=1,30
      M=N-1
      MM=2*M
      MMM=2*M+1
      IF(N .EQ. 1) GO TO 5
      TERM=(-1)*(MU-(2*MM-1)**2)*(MU-(2*MM+1)**2)/(MMM*(MMM-1)*RHO**2)*
     $      TERM
      IF(NPRINT .EQ. 1) PRINT 100,TERM
      Q=Q+TERM
      IF(ZABS(TERM) .GE. ZABS(TERMS) .OR. ZABS(TERM) .LE. 1.D-17)
     $ GO TO 6
```

```
      TERMS=TERM
      GO TO 4
5     TERM=(MU-1)/RHO
      Q=TERM
      IF(NPRINT .EQ. 1) PRINT 100,TERM
      TERMS=TERM
4     CONTINUE
6     CONTINUE
      IF(NPRINT .EQ. 1) PRINT 300,Q
300   FORMAT(1H0,'Q=',2D30.15)
      BJ=RT*ZCOS(Z-K*PI/2.-PI/4.)*P-RT*ZSIN(Z-K*PI/2.-PI/4.)*Q
      IF(KIND .EQ. 1) GO TO 8
      BY=RT*ZSIN(Z-K*PI/2.-PI/4.)*P+RT*ZCOS(Z-K*PI/2.-PI/4.)*Q
8     RETURN
C
C
C  POWER SERIES EXPANSION
C
10    NTERMS=35
      KFAC=1
      IF(K .LE. 1) GO TO 30
      DO 20 N=2,K
20    KFAC=KFAC*N
30    TERM=(Z/2.)**K/KFAC
      BJ=TERM
      IF(KIND .EQ. 1) GO TO 91
      DG1=0.
      DG2=0.
      IF(K .EQ. 0) GO TO 80
      DO 60 N=1,K
60    DG2=DG2+1./N
80    TSUM3=-TERM*DG2
      SUMT3=TSUM3
91    DO 40 M=1,NTERMS
      TERM=TERM*(Z/2.)**2*(-1)/((K+M)*M)
      BJ=BJ+TERM
      IF(KIND .EQ. 1) GO TO 92
      DG1=DG1+1.D0/M
      DG2=DG2+1.D0/(M+K)
      DGMSUM=DG1+DG2
      TSUM3=-TERM*DGMSUM
      SUMT3=SUMT3+TSUM3
92    IF(ZABS(TERM) .LE. 1.D-17) GO TO 50
40    CONTINUE
50    IF(KIND .EQ. 1) GO TO 93
      TERM3=SUMT3/PI
      TERM1=(2./PI)*BJ*(EULER+ZLOG(Z/2.))
      SUMT2=(0.,0.)
      IF(K .EQ. 0) GO TO 120
      KM1FAC=KFAC/K
      TSUM2=KM1FAC/((Z/2.)**K)
      SUMT2=TSUM2
      IF(K .EQ. 1) GO TO 120
      KM1=K-1
      DO 130 M=1,KM1
      KMM=K-M
      TSUM2=TSUM2/(KMM*M)*(Z/2.)**2
130   SUMT2=SUMT2+TSUM2
```

```
120    TERM2=-SUMT2/PI
       BY=TERM1+TERM2+TERM3
93     RETURN
       END
```

```
      SUBROUTINE HJFUNC(ARG,BJ,BJP,H2,H2P)
C
C     SPECIAL PURPOSE ROUTINE FOR CALCULATING BESSEL FUNCTIONS(JM,VM),
C     HANKEL FUNCTIONS(H2) AND THEIR DERIVATIVES FOR COMPLEX ARGUMENT.
C     REQUIRES FOR STARTING CONDITIONS AN INDEPENDENT CALCULATION OF
C     J0,Y0,J1 AND Y1.  THE LATTER ARE OBTAINED FROM SUBROUTINE CBESJY.
C
      IMPLICIT COMPLEX*16(A-H,O-Z)
      COMPLEX*16 IM
      REAL*8 PI
      PARAMETER (MMAX=51)
      PARAMETER (MPLUS=MMAX+30)
      DIMENSION BJ(0:MMAX),BJP(0:MMAX),H2(0:MMAX),H2P(0:MMAX),BY(0:MMAX)
      DATA IM/(0.0,1.0)/
      DATA PI/3.14159265358979D0/
C
C     CALCULATE BY
      M=0
      CALL CBESJY(ARG,M,BJEXAC,BY(M),0,0)
      M=1
      CALL CBESJY(ARG,M,BJTEMP,BY(M),0,0)
      DO 100 M=2,MMAX
      BY(M)=2.0*(M-1)/ARG*BY(M-1)-BY(M-2)
100   CONTINUE
C
C     CALCULATE BJ
      BJP2=0.0
      BJP1=1.0
      DO 150 M=MPLUS-2,MMAX-1,-1
      BJTEMP=2.0*(M+1)/ARG*BJP1-BJP2
      BJP2=BJP1
      BJP1=BJTEMP
      IF(ZABS(BJP1) .GT. 1.D20) THEN
         BJP2=BJP2/BJP1
         BJP1=1.0
      ENDIF
150   CONTINUE
C
      BJ(MMAX)=BJP2
      BJ(MMAX-1)=BJP1
C
      DO 200 M=MMAX-2,0,-1
      BJ(M)=2.0*(M+1)/ARG*BJ(M+1)-BJ(M+2)
      IF(ZABS(BJ(M)) .GT. 1.0D20) THEN
         BJSAVE=BJ(M)
         DO 199 MM=M,MMAX
199      BJ(MM)=BJ(MM)/BJSAVE
      ENDIF
200   CONTINUE
      TERM=BJEXAC/BJ(0)
      DO 300 M=0,MMAX
      BJ(M)=TERM*BJ(M)
300   CONTINUE
C
C     CALCULATE H2
      DO 350 M=0,MMAX
      H2(M)=BJ(M)-IM*BY(M)
350   CONTINUE
```

```
C
C       CALCULATE BJP AND H2P
        BJP(0)=-BJ(1)
        H2P(0)=-H2(1)
        DO 400 M=1,MMAX-1
        BJP(M)=0.5*(BJ(M-1)-BJ(M+1))
        H2P(M)=0.5*(H2(M-1)-H2(M+1))
400     CONTINUE
C
        WEXACT=2.0/(PI*ARG)
        WMEQ0=BJ(1)*BY(0)-BJ(0)*BY(1)
        WMEQMX=BJ(MMAX)*BY(MMAX-1)-BJ(MMAX-1)*BY(MMAX)
        IF(ZABS((WMEQ0 -WEXACT)/WEXACT) .GT. 1.0D-4 .OR.
       $    ZABS((WMEQMX-WEXACT)/WEXACT) .GT. 1.0D-4)then
            write(44,*)' WARNING-WRONSKIAN EXCEEDS TOLERANCE'
            write(44,*)'ARG =',ARG
            write(44,*)'W0 =',ZABS((WMEQ0 -WEXACT)/WEXACT)
            write(44,*)'WMX=',ZABS((WMEQMX-WEXACT)/WEXACT)
        end if
C
        RETURN
        END
```

```
      SUBROUTINE CLIN EQ (A, B, X, N, N DIM, IFLAG, ERR)
C
C     CLIN EQ USES L-U DECOMPOSITION TO FIND THE TRIANGULAR MATRICES L
C     AND U SUCH THAT L * U = A.  THE MATRICES L AND U ARE STORED IN A.
C     THIS FORM IS USED WITH BACK SUBSTITUTION TO FIND THE SOLUTION X OF
C     A * X = L * U * X = B.  N IS THE NUMBER OF EQUATIONS AND NDIM IS
C     THE DIMENSION OF ALL ARRAYS.  ERR IS THE ESTIMATED RELATIVE ERROR
C     OF THE SOLUTION VECTOR.
C
C     IF IFLAG = 0, THEN L, U AND X ARE COMPUTED.  OTHERWISE, IT IS
C     ASSUMED THAT L AND U HAVE BEEN COMPUTED IN A PREVIOUS CALL AND ARE
C     STILL STORED IN A.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMPLEX*16  A, B, X, T
      INTEGER*2 IROW
      DIMENSION  A(N DIM, N DIM), B(N DIM), X(N DIM)
      DIMENSION  IROW(51), Q(51)
      DATA  EPS /1.0D-15/
C
      IF(N .GT. 51)  GO TO 900
      IF(N .EQ. 1) GO TO 850
      IF(IFLAG .NE. 0)  GO TO 600
      DO 50  I = 1,N
      Q(I) = 0.0D0
      DO 40  J = 1,N
      QQ = ZABS(A(I,J))
   40 IF(Q(I) .LT. QQ)  Q(I) = QQ
      IF(Q(I) .EQ. 0.0D0)  GO TO 901
   50 CONTINUE
      ERR = EPS
      PPIV = 0.0D0
      DO 100  I = 1,N
  100 IROW(I) = I
C
      DO 500  L = 1,N
      PIVOT = 0.0D0
      K = L - 1
      DO 240  I = L,N
      IF(K .LT. 1)  GO TO 230
      DO 220  J = 1,K
  220 A(I,L) = A(I,L) - A(J,L) * A(I,J)
  230 F = ZABS(A(I,L)) / Q(I)
      IF(PIVOT .GT. F)  GO TO 240
      PIVOT = F
      NPIVOT = I
  240 CONTINUE
      IF(PIVOT .EQ. 0.0D0)  GO TO 901
      IF(PPIV .LE. PIVOT)  GO TO 250
      ERR = ERR * PPIV / PIVOT
c      PRINT *,'ERR=',ERR
c      write(44,*)'err=',err
      IF(ERR.GE.1.0D0)  GO TO 901
  250 PPIV = PIVOT
      IF(NPIVOT .EQ. L)  GO TO 280
      Q(NPIVOT) = Q(L)
      J = IROW(L)
      IROW(L) = IROW(NPIVOT)
```

```
      IROW(NPIVOT) = J
      DO 260  I = 1,N
      T = A(L,I)
      A(L,I) = A(NPIVOT,I)
      A(NPIVOT,I) = T
  260 CONTINUE
  280 IF(L .EQ. N)  GO TO 500
      T = (1.0D0,0.0D0) / A(L,L)
      K = L + 1
      M = L - 1
      DO 450  I = K,N
      IF(M .LT. 1)  GO TO 400
      DO 350  J = 1,M
  350 A(L,I) = A(L,I) - A(L,J) * A(J,I)
  400 A(L,I) = T * A(L,I)
  450 CONTINUE
  500 CONTINUE
      IF(ERR .GT. 1.0D-5)   PRINT 998, ERR
C
  600 DO 620  I = 2,N
  620 X(I) = (0.0D0,0.0D0)
      J = IROW(1)
      X(1) = B(J) / A(1,1)
      DO 700  I = 2,N
      J = IROW(I)
      K = I - 1
      DO 650  L = 1,K
  650 X(I) = X(I) + A(I,L) * X(L)
      X(I) = (B(J) - X(I)) / A(I,I)
  700 CONTINUE
      K = N - 1
      DO 800  I = 1,K
      J = N - I
      M = J + 1
      DO 800  L = M,N
      X(J) = X(J) - X(L) * A(J,L)
  800 CONTINUE
      RETURN
C
  850 X(1)=B(1)/A(1,1)
      RETURN
C
  900 PRINT 999
      ERR = 1.0D0
      RETURN
  901 PRINT 997
      ERR = 1.0D0
      RETURN
  997 FORMAT('0ERROR IN CLIN EQ, MATRIX IS SINGULAR')
  998 FORMAT('0CAUTION-',
     $       ' CLIN EQ HAS DECOMPOSED AN ILL-CONDITIONED MATRIX.'/
     $       ' RESULTS WILL HAVE RELATIVE ERROR =',1PE12.2)
  999 FORMAT('0ERROR IN CLIN EQ, MATRIX SIZE GREATER THAN 51')
      END
```

APPENDIX B

PLOTTING ROUTINE LISTING

```
C     MAIN ROUTINE FOR PROGRAM
C
      complex*16 c1,c2
      real*8 frqkhz,delta1,delta2,xyint,x0,y0,r,alpha
      real*4 xplot,yplot1,yplot2,yplot3,yplot4,yplot5,yplot6
      COMMON/COM1/xplot(302),yplot1(302),yplot2(302),yplot3(302),
     $                        yplot4(302),yplot5(302),yplot6(302)
      COMMON/COM2/frqkhz,delta1,delta2,c1,c2,xyint,alpha
      common/com3/iflag,nrslab,numpts
      common/com4/x0,y0,r
      CHARACTER*1 ANSWER
      character*20 fn

      call init
C
C
      print *,'Enter name of file containing data to be plotted:
      read(5,850) fn
850   format(a20)
      OPEN(UNIT=25,file=fn,status='OLD',form='unformatted')
      READ(25) iflag,nrslab,frqkhz,delta1,delta2,c1,c2,xyint,alpha
      if(iflag .eq. 0) then
         READ(25) x0,y0,numpts
      else
         READ(25) r,numpts
      endif
      icount=0
      do i=1,numpts
      read(25,end=900) xplot(i),yplot1(i),yplot2(i),yplot3(i),yplot4(i),
     $                          yplot5(i),yplot6(i)
      xplot(i)=xplot(i)/1.0e3
      icount=icount+1
      enddo
C
C
900   numpts=icount
      CALL PLTDTA
C
999   STOP
      END
```

```
subroutine init
common/com5/ sizex,xmin,xmax,xtic,sizey,ymin,ymax,ytic
sizex=6.00
xmin=-5.0
xmax=7.0
xtic=2.0
sizey=6.00
ymin=-10.0
ymax=4.0
ytic=2.0
return
end
```

```
      SUBROUTINE PLTBGN
      open(unit=20,file='/dev/plt7550a')
      call hpinit(2,0,0,0,20)
      RETURN
      END

      SUBROUTINE PLTDTA
      complex*16 c1,c2
      real*8 frqkhz,delta1,delta2,xyint,x0,y0,r,alpha
      COMMON/COM1/xplot(302),yplot1(302),yplot2(302),yplot3(302),
     $                      yplot4(302),yplot5(302),yplot6(302)
      COMMON/COM2/frqkhz,delta1,delta2,c1,c2 xyint,alpha
      common/com3/iflag,nrslab,numpts
      common/com4/x0,y0,r
      common/com5/ sizex,xmin,xmax,xtic,sizey,ymin,ymax,ytic
      LOGICAL UP(302),UPL(2)
      DIMENSION XL(2),YL(2),XLAB0(3),xlab1(7),YLABa(5),ylabp(6)
      dimension ia(8)
      character*1 answer
C
      data(XLAB0(j),j= 1,3)/4HRANG,4HE (M,4Hm)  /
      data(XLAB1(j),j= 1,7)/4HDIST,4HURBA,4HNCE ,4HLOCA,4HTION,4H-S (,
     $                       4HMm) /
      data(YLABa(j),j= 1,5)/4HAMPL,4HITUD,4HE RA,4HTIO-,4HdB  /
      data(YLABp(j),j= 1,6)/4HPHAS,4HE DI,4HFFER,4HENCE,4H - D,4HEG  /
C
      data ia/82,79,57,48,73,87,73,80/
c              R  O  9  0  I  W  I  P
C
      HEIGHT=0.1
      DO 100 J=1,302
      UP(J)=.FALSE.
100   CONTINUE
      DO 110 J=1,2
      UPL(J)=.FALSE.
110   CONTINUE
C
C     DRAW AMPLITUDE PLOT
      print *,'Do you want amplitude plots?'
      read (5,972) answer
972   format(a1)
      if(answer .eq. 'Y' .or. answer .eq. 'y') then
      print *
      print *
      print *,'How many amplitude plots do you want?'
      read *, numplt
      print *,'numplt=',numplt
      print *
      if(numplt .eq. 1) then
         print *
         print *,'Enter appropriate code for desired plot'
         print *
         print *,'     1      ez plot'
         print *,'     2      hy plot'
         print *,'     3      hr plot'
         read *,iplot
      else
         if(numplt .eq. 2) then
```

```
            print *
            print *,'Enter appropriate code for desired plots'
            print *
            print *,'      1      ez and hy plots'
            print *,'      2      ez and hr plots'
            print *,'      3      hy and hr plots'
            read *,iplot
         endif
      endif
c
      call input
      CALL PLTBGN
      call buff(1,ia,xbuf,8)
      CALL PLOT(1.0,1.5,-3)
      CALL BORDER(sizex,xmin,xmax,xtic,1,sizey,ymin,ymax,ytic,1)
      xinc=(xmax-xmin)/sizex
      yinc=(ymax-ymin)/sizey
      XL(1)=1.2
      XL(2)=2.0
      if((numplt .eq. 1 .and. iplot .eq. 1) .or.
     $   (numplt .eq. 2 .and. (iplot .eq. 1 .or. iplot .eq. 2)) .or.
     $   (numplt .eq. 3)) then
          CALL CURVE(xPLOT,yplot1,UP,numpts,xmin,ymin,xinc,yinc,1)
          CALL SYMBOL(0.0,-0.6,HEIGHT,8H|EZ/EZU|,0.0,8)
          YL(1)=-0.6
          YL(2)=-0.6
          CALL CURVE(XL,YL,UPL,2,0.0,0.0,1.0,1.0,1)
      endif
      if((numplt .eq. 1 .and. iplot .eq. 2) .or.
     $   (numplt .eq. 2 .and. (iplot .eq. 1 .or. iplot .eq. 3)) .or.
     $   (numplt .eq. 3)) then
          CALL CURVE(xPLOT,yplot2,UP,numpts,xmin,ymin,xinc,yinc,4)
          CALL SYMBOL(0.0,-0.8,HEIGHT,8H|HY/HYU|,0.0,8)
          YL(1)=-0.8
          YL(2)=-0.8
          CALL CURVE(XL,YL,UPL,2,0.0,0.0,1.0,1.0,4)
      endif
      if((numplt .eq. 1 .and. iplot .eq. 3) .or.
     $   (numplt .eq. 2 .and. (iplot .eq. 2 .or. iplot .eq. 3)) .or.
     $   (numplt .eq. 3)) then
          CALL CURVE(xPLOT,yplot3,UP,numpts,xmin,ymin,xinc,yinc,5)
          CALL SYMBOL(0.0,-1.0,HEIGHT,10H|HR/HRU|  ,0.0,10)
          YL(1)=-1.0
          YL(2)=-1.0
          CALL CURVE(XL,YL,UPL,2,0.0,0.0,1.0,1.0,5)
      endif
C
      if(iflag .eq. 0) then
         xpos=(sizex-1.2)/2.0
         CALL SYMBOL(xpos,-0.30,HEIGHT,XLAB0,0.,12)
      else
         xpos=(sizex-2.6)/2.0
         CALL SYMBOL(xpos,-0.30,HEIGHT,XLAB1,0.,28)
      endif
      ypos=(sizey-1.8)/2.0
      CALL SYMBOL(-0.30,ypos,HEIGHT,YLABa,90.,20)
C
C     LABEL PLOTS
```

```
      CALL SYMBOL(3.0,-0.6,HEIGHT,20HFREQUENCY-       HZ,0.0,20)
      CALL NUMBER(4.0,-0.6,HEIGHT,sngl(frqkhz*1.0e3),0.0,1)
      if(iflag .eq. 1) then
        call symbol(3.0,-0.8,height,20hRANGE-           Mm,0.0,20)
        call number(3.7,-0.8,height,sngl(r/1.0e3),0.0,3)
      endif
      call symbol(0.0,sizey+0.9,height,20hC1=(       ,       ),0.0,20)
      c1r=dble(c1)
      c1i=dimag(c1)
      CALL NUMBER(0.5,sizey+0.9,HEIGHT,c1r,0.0,2)
      CALL NUMBER(1.3,sizey+0.9,HEIGHT,c1i,0.0,2)
      call symbol(0.0,sizey+0.7,height,20hC2=(       ,       ),0.0,20)
      c2r=dble(c2)
      c2i=dimag(c2)
      CALL NUMBER(0.5,sizey+0.7,HEIGHT,c2r,0.0,2)
      CALL NUMBER(1.3,sizey+0.7,HEIGHT,c2i,0.0,2)
      if(iflag .eq. 0) then
        call symbol(0.0,sizey+0.5,height,14hX0=        KM,0.0,14)
        CALL NUMBER(0.4,sizey+0.5,HEIGHT,sngl(x0),0.0,2)
        call symbol(3.0,sizey+0.5,height,14hY0=        KM,0.0,14)
        CALL NUMBER(3.4,sizey+0.5,HEIGHT,sngl(y0),0.0,2)
      else
        call symbol(0.0,sizey+0.5,height,20hALPHA=           DEG,0.0,20)
        CALL NUMBER(0.7,sizey+0.5,HEIGHT,sngl(alpha),0.0,2)
        if(alpha .eq. 0.0) then
          call symbol(3.0,sizey+0.5,height,22hY-INTERCEPT=       KM,
     $            0.0,22)
        else
          call symbol(3.0,sizey+0.5,height,22hX-INTERCEPT=       KM,
     $            0.0,22)
        endif
        CALL NUMBER(4.25,sizey+0.5,HEIGHT,sngl(xyint),0.0,2)
      endif
      call symbol(0.0,sizey+0.3,height,20hNRSLAB=          ,0.0,20)
      CALL NUMBER(0.8,sizey+0.3,HEIGHT,real(nrslab),0.0,-1)
      call symbol(3.0,sizey+0.9,height,22hDELTA1=          DEG,0.0,22)
      CALL NUMBER(4.0,sizey+0.9,HEIGHT,sngl(delta1),0.0,2)
      call symbol(3.0,sizey+0.7,height,22hDELTA2=          DEG,0.0,22)
      CALL NUMBER(4.0,sizey+0.7,HEIGHT,sngl(delta2),0.0,2)
      CALL PLTEND
      endif
C
C     DRAW SIGNAL LEVEL PLOT
      print *,'Do you want phase plots?'
      read (5,972) answer
      if(answer .eq. 'Y' .or. answer .eq. 'y') then
      print *
      print *
      print *,'How many phase plots do you want?'
      read *, numplt
      print *,'numplt=',numplt
      print *
      if(numplt .eq. 1) then
         print *
         print *,'Enter appropriate code for desired plot'
         print *
         print *,'      1      ez plot'
         print *,'      2      hy plot'
```

```
            print *,'      3      hr plot'
            read *,iplot
         else
            if(numplt .eq. 2) then
               print *
               print *,'Enter appropriate code for desired plots'
               print *
               print *,'      1      ez and hy plots'
               print *,'      2      ez and hr plots'
               print *,'      3      hy and hr plots'
               read *,iplot
            endif
         endif
c
         call input
         CALL PLTBGN
         call buff(1,ia,xbuf,8)
         CALL PLOT(1.0,1.5,-3)
         CALL BORDER(sizex,xmin,xmax,xtic,1,sizey,ymin,ymax,ytic,1)
         xinc=(xmax-xmin)/sizex
         yinc=(ymax-ymin)/sizey
         XL(1)=1.4
         XL(2)=2.2
         if((numplt .eq. 1 .and. iplot .eq. 1) .or.
        $   (numplt .eq. 2 .and. (iplot .eq. 1 .or. iplot .eq. 2)) .or.
        $   (numplt .eq. 3)) then
             CALL CURVE(xPLOT,yplot4,UP,numpts,xmin,ymin,xinc,yinc,1)
             CALL SYMBOL(0.0,-0.6,HEIGHT,13HPHASE(EZU/EZ),0.0,13)
             YL(1)=-0.6
             YL(2)=-0.6
             CALL CURVE(XL,YL,UPL,2,0.0,0.0,1.0,1.0,1)
         endif
         if((numplt .eq. 1 .and. iplot .eq. 2) .or.
        $   (numplt .eq. 2 .and. (iplot .eq. 1 .or. iplot .eq. 3)) .or.
        $   (numplt .eq. 3)) then
             CALL CURVE(xPLOT,yplot5,UP,numpts,xmin,ymin,xinc,yinc,4)
             CALL SYMBOL(0.0,-0.8,HEIGHT,13HPHASE(HYU/HY),0.0,13)
             YL(1)=-0.8
             YL(2)=-0.8
             CALL CURVE(XL,YL,UPL,2,0.0,0.0,1.0,1.0,4)
         endif
         if((numplt .eq. 1 .and. iplot .eq. 3) .or.
        $   (numplt .eq. 2 .and. (iplot .eq. 2 .or. iplot .eq. 3)) .or.
        $   (numplt .eq. 3)) then
             CALL CURVE(xPLOT,yplot6,UP,numpts,xmin,ymin,xinc,yinc,5)
             CALL SYMBOL(0.0,-1.0,HEIGHT,13HPHASE(HRU/HR),0.0,13)
             YL(1)=-1.0
             YL(2)=-1.0
             CALL CURVE(XL,YL,UPL,2,0.0,0.0,1.0,1.0,5)
         endif

         if(iflag .eq. 0) then
            xpos=(sizex-1.2)/2.0
            CALL SYMBOL(xpos,-0.30,HEIGHT,XLAB0,0.,12)
         else
            xpos=(sizex-2.6)/2.0
            CALL SYMBOL(xpos,-0.30,HEIGHT,XLAB1,0.,28)
         endif
```

```
      ypos=(sizey-2.1)/2.0
      CALL SYMBOL(-0.30,ypos,HEIGHT,YLABp,90.,24)
C
C     LABEL PLOTS
      CALL SYMBOL(3.0,-0.6,HEIGHT,20HFREQUENCY=         HZ,0.0,20)
      CALL NUMBER(4.0,-0.6,HEIGHT,sngl(frqkhz*1.0e3),0.0,1)
      if(iflag .eq. 1) then
        call symbol(3.0,-0.8,height,20hRANGE=              Mm,0.0,20)
        call number(3.7,-0.8,height,sngl(r/1.0e3),0.0,3)
      endif
      call symbol(0.0,sizey+0.9,height,20hC1=(        ,        ),0.0,20)
      c1r=dble(c1)
      c1i=dimag(c1)
      CALL NUMBER(0.5,sizey+0.9,HEIGHT,c1r,0.0,2)
      CALL NUMBER(1.3,sizey+0.9,HEIGHT,c1i,0.0,2)
      call symbol(0.0,sizey+0.7,height,20hC2=(        ,        ),0.0,20)
      c2r=dble(c2)
      c2i=dimag(c2)
      CALL NUMBER(0.5,sizey+0.7,HEIGHT,c2r,0.0,2)
      CALL NUMBER(1.3,sizey+0.7,HEIGHT,c2i,0.0,2)
      if(iflag .eq. 0) then
        call symbol(0.0,sizey+0.5,height,14hX0=         KM,0.0,14)
        CALL NUMBER(0.4,sizey+0.5,HEIGHT,sngl(x0),0.0,2)
        call symbol(3.0,sizey+0.5,height,14hY0=         KM,0.0,14)
        CALL NUMBER(3.4,sizey+0.5,HEIGHT,sngl(y0),0.0,2)
      else
        call symbol(0.0,sizey+0.5,height,20hALPHA=            DEG,0.0,20)
        CALL NUMBER(0.7,sizey+0.5,HEIGHT,sngl(alpha),0.0,2)
        if(alpha .eq. 0.0) then
          call symbol(3.0,sizey+0.5,height,22hY-INTERCEPT=        KM,
     $                0.0,22)
        else
          call symbol(3.0,sizey+0.5,height,22hX-INTERCEPT=        KM,
     $                0.0,22)
        endif
        CALL NUMBER(4.25,sizey+0.5,HEIGHT,sngl(xyint),0.0,2)
      endif
      call symbol(0.0,sizey+0.3,height,20hNRSLAB=             ,0.0,20)
      CALL NUMBER(0.8,sizey+0.3,HEIGHT,real(nrslab),0.0,-1)
      call symbol(3.0,sizey+0.9,height,22hDELTA1=           DEG,0.0,22)
      CALL NUMBER(4.0,sizey+0.9,HEIGHT,sngl(delta1),0.0,2)
      call symbol(3.0,sizey+0.7,height,22hDELTA2=           DEG,0.0,22)
      CALL NUMBER(4.0,sizey+0.7,HEIGHT,sngl(delta2),0.0,2)
      CALL PLTEND
        endif

      RETURN
      END

      SUBROUTINE PLTEND
      CALL NEWPEN(0)
      CALL PLOT(0.,0.,999)
      RETURN
      END
```

```
      subroutine input
      common/com5/ sizex,xmin,xmax,xtic,sizey,ymin,ymax,ytic
      character*1 answer

      print *
      print *,'Current values for sizex and sizey are:',sizex,sizey
      print *,'Do you want to change them?'
      read (5,972) answer
972   format(a1)
      if(answer .eq. 'y' .or. answer .eq. 'Y') then
         print *,'Enter values for sizex,sizey'
         read *,sizex,sizey
         print *,'size=',sizex,sizey
      endif

      print *
      print *,'Current values for xmin and xmax are:',xmin,xmax
      print *,'Do you want to change them?'
      read (5,972) answer
      if(answer .eq. 'y' .or. answer .eq. 'Y') then
         print *,'Enter values for xmin,xmax'
         read *,xmin,xmax
         print *,'xmin,xmax=',xmin,xmax
      endif

      print *
      print *,'Current values for ymin and ymax are:',ymin,ymax
      print *,'Do you want to change them?'
      read (5,972) answer
      if(answer .eq. 'y' .or. answer .eq. 'Y') then
         print *,'Enter values for ymin,ymax'
         read *,ymin,ymax
         print *,'ymin,ymax=',ymin,ymax
      endif

      print *
      print *,'Tic marks will be every ',xtic,' units on the x-axis and'
      print *,'                  every ',ytic,' units on the y-axis.'
      print *,'Do you want to change them?'
      read (5,972) answer
      if(answer .eq. 'y' .or. answer .eq. 'Y') then
         print *,'Enter values for xtic,ytic'
         read *,xtic,ytic
         print *,'xtic,ytic=',xtic,ytic
      endif
      return
      end
```

```
      subroutine border(xlng,xmin,xmax,xinc,nx,ylng,ymin,ymax,yinc,ny)
c
      dimension xinc(nx),yinc(ny)
      logical fy,fx
c
      fx=.false.
      fy=.false.
      if(nx .eq. 1) fx=.true.
      if(ny .eq. 1) fy=.true.
      xt=xlng-.1
      yt=ylng-.1
      xscale=xlng/(xmax-xmin)
      yscale=ylng/(ymax-ymin)
      ym=abs(ymin)
      yln=-.4
      if(ym .ge. 10.) yln=yln-.1
      if(ym .ge. 100.) yln=yln-.1
      if(ym .ge. 1000.) yln=yln-.1
      if(ymin .lt. 0.) yln=yln-.1
      ym=abs(ymax)
      ylm=-.4
      if(ym .ge. 10.) ylm=ylm-.1
      if(ym .ge. 100.) ylm=ylm-.1
      if(ym .ge. 1000.) ylm=ylm-.1
      if(ymax .lt. 0.) ylm=ylm-.1
      xm=abs(xmax)
      xlm=-.3
      if(xm .ge. 10.) xlm=xlm-.1
      if(xm .ge. 100.) xlm=xlm-.1
      if(xm .ge. 1000.) xlm=xlm-.1
      if(xmax .lt. 0.) xlm=xlm-.1
      if(fx) dx=xinc(1)
      if(fy) dy=yinc(1)
      iy=1
      yl=0.
      call number(yln,0.,.1,ymin,0.,1)
      call plot(0.,0.,3)
      if(fy) go to 110
10    yp=(yinc(iy)-ymin)*yscale
      go to 111
110   yl=yl+dy
      yp=yl*yscale
111   if(yp .lt.   0.) go to 99
      if(yp .ge. ylng) go to 11
      call plot(0.,yp,2)
      call plot(.1,yp,2)
      call plot(0.,yp,2)
      if(fy) go to 110
      iy=iy+1
      if(iy .le. ny) go to 10
11    call plot(0.,ylng,2)
      call number(ylm,ylng-.1,.1,ymax,0.,1)
      call plot(0.,ylng,3)
      ix=1
      xl=0.
      if(fx) go to 112
12    xp=(xinc(ix)-xmin)*xscale
      go to 120
```

```
112   xl=xl+dx
      xp=xl*xscale
120   if(xp .lt.   0.) go to 99
      if(xp .ge. xlng) go to 13
      call plot(xp,ylng,2)
      call plot(xp,yt,2)
      call plot(xp,ylng,2)
      if(fx) go to 112
      ix=ix+1
      if(ix .le. nx) go to 12
13    call plot(xlng,ylng,2)
      if(fy) go to 130
113   iy=iy-1
      if(iy .le. 0) go to 15
      yp=(yinc(iy)-ymin)*yscale
      go to 14
130   yl=yl-dy
      yp=yl*yscale
      if(yp .le. 0.) go to 15
14    call plot(xlng,yp,2)
      call plot(xt,yp,2)
      call plot(xlng,yp,2)
      if(fy) go to 130
      go to 113
15    call plot(xlng,0.,2)
      call number(xlng+xlm,-.2,.1,xmax,0.,1)
      call plot(xlng,0.,3)
      if(fx) go to 150
115   ix=ix-1
      if(ix .le. 0) go to 17
      xp=(xinc(ix)-xmin)*xscale
      go to 16
150   xl=xl-dx
      xp=xl*xscale
      if(xp .le. 0.) go to 17
16    call plot(xp,0.,2)
      call plot(xp,.1,2)
      call plot(xp,0.,2)
      if(fx) go to 150
      go to 115
17    call plot(0.,0.,2)
      call number(0.,-.2,.1,xmin,0.,1)
      return
99    print 100,xlng,xmin,xmax,xinc(1),nx,ylng,ymin,ymax,yinc(1),ny
100   format('0***  Error in BORDER:  xlng, xmin, xmax, xinc(1), nx =',
     $       1p4e15.5,i5/24x,'ylng, ymin, ymax, yinc(1), ny =',1p4e15.5,
     $       i5/'0***')
      call pltend
      stop
      end
```

```
      subroutine curve(x,y,up,nrpts,xmin,ymin,xinc,yinc,line)
c
c x,y,up must be dimensioned at least nrpts
c xmin,ymin are x,y origin in user units
c xinc,yinc are x,y scales in user units per inch
c
c line=1:  solid
c      2:  long dash
c      3:  medium dash
c      4:  short dash
c      5:  dotted
c      6:  short + long dash
c      7:  short + short + long dash
c
      logical up,up1,up2
      dimension ipen(8),joc(7),x(nrpts),y(nrpts),up(nrpts)
      data ipen/2,2,2,3,2,3,2,3/,joc/18, 11, 14, 23, 32, 41, 16/
      data delr/.1/
c
      if(nrpts .le. 1) go to 99
c
      if(line) 1,2,3
1     kk=mod(line,7)+7
      go to 4
2     kk=0
      go to 4
3     kk=mod(line,7)
4     kk=kk+1
      jo=joc(kk)/10
      jc=joc(kk)-10*jo
      ip=ipen(jo)
c
      j=0
      dr=0.
      rho1=0.
      rho2=delr
      px1=(x(1)-xmin)/xinc
      py1=(y(1)-ymin)/yinc
      up1=up(1)
      if(.not. up1) then
c
c go to first position with pen up
        call plot(px1,py1,3)
        if(kk .eq. 6) then
          px2=(x(2)-xmin)/xinc
          py2=(y(2)-ymin)/yinc
          delx=px2-px1
          dely=py2-py1
          rho=sqrt(delx**2+dely**2)
          if(rho .eq. 0.) then
            dx 6=delx*.1
            dy 6=dely*.1
          else
            dx 6=delx*delr/rho*.1
            dy 6=dely*delr/rho*.1
          end if
          call plot(px1+dx6,py1+dy6,2)
        end if
```

```
      end if
c
      do 40 i=2,nrpts
      px2=(x(i)-xmin)/xinc
      py2=(y(i)-ymin)/yinc
      up2=up(i)
      if(up2) then
        dr=0.
        rho1=0.
        rho2=delr
        go to 39
      end if
      if(up1) then
c pen has been up, prepare to lower pen
        call plot(px2,py2,3)
        go to 39
      end if
      if(kk .eq. 2) go to 38
      delx=px2-px1
      dely=py2-py1
      rho=sqrt(delx**2+dely**2)
      rho1=rho1+rho
      if(rho2 .gt. rho1) go to 38
      delx=delx*delr/rho
      dely=dely*delr/rho
      dx 6=delx*.1
      dy 6=dely*.1
      if(dr .eq. 0.) go to 20
      dx=delx*dr/delr
      dy=dely*dr/delr
      px1=px1+dx
      py1=py1+dy
      go to 21
20    if(rho2 .gt. rho1) go to 38
      px1=px1+delx
      py1=py1+dely
21    call plot(px1,py1,ip)
      if(kk .eq. 6) call plot(px1+dx6,py1+dy6,2)
      j=j+1
      ip=ipen(jo+mod(j,jc))
      rho2=rho2+delr
      go to 20
38    call plot(px2,py2,ip)
      dr=rho2-rho1
39    px1=px2
      py1=py2
      up1=up2
40    continue
99    return
      end
```

END
DATE
FILMED
11-88
DTIC